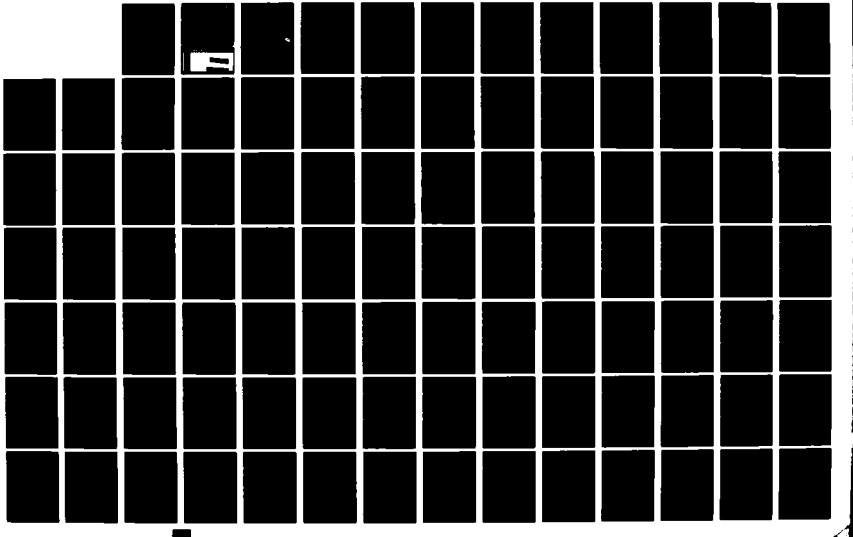
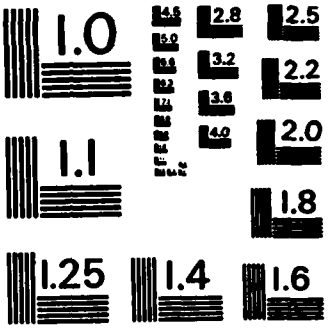
AD-A121 058 COMMON MODULAR MULTIMODE RADAR (CMMR) SOFTWARE ACQUISITION STUDY(U) ARINC RESEARCH CORP ANNAPOLIS MD S BAILY ET AL. MAR 81 2302-01-1-2291 F09603-80-G-3338 1/2

UNCLASSIFIED F/G 15/5 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS - 1963 - A

Publication 2302-01-1-2291

FINAL REPORT

COMMON MODULAR MULTIMODE RADAR (CMMR) SOFTWARE ACQUISITION STUDY

March 1981

Prepared for
DEPUTY FOR RECONNAISSANCE AND
ELECTRONIC WARFARE SYSTEMS (ASD/RW)
AERONAUTICAL SYSTEMS DIVISION
WRIGHT-PATTERSON AIR FORCE BASE, DAYTON, OHIO 45433
under Contract F09603-80-G-3338

ARINC RESEARCH CORPORATION

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: 2302-01-1-2291	2. GOVT ACCESSION NO.: AD-A121058
3. RECIPIENT'S CATALOG NUMBER:	
4. TITLE (and Subtitle): Common Modular Multimode Radar (CMMR) Software Acquisition Study	5. TYPE OF REPORT & PERIOD COVERED:
	6. PERFORMING ORG. REPORT NUMBER: 2302-01-1-2291
7. AUTHOR(s): S. Baily, E. Straub, R. Gilbertson	8. CONTRACT OR GRANT NUMBER(s): F09603-80-G-3338
9. PERFORMING ORGANIZATION NAME AND ADDRESS: ARINC Research Corp. 2551 Riva Road Annapolis, Md. 21401	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:
11. CONTROLLING OFFICE NAME AND ADDRESS: Deputy for Reconnaissance and Electronic Warfare Systems (ASD/RW), Aeronautical Systems Division Wright-Patterson AFB, Ohio 45433	12. REPORT DATE: March 1981
	13. NUMBER OF PAGES: 129
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)

Unlimited

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

DTIC ELECTE NOV 3 1982 H

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Software Acquisition
Radar

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This document reports on ARINC Research Corporation's work in developing and evaluating alternative acquisition strategies for the USAF's Common Modular Multimode Radar (CMMR) program.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Ent

FINAL REPORT

COMMON MODULAR MULTIMODE RADAR (CMMR) SOFTWARE ACQUISITION STUDY

March 1981

Prepared for
Deputy for Reconnaissance and Electronic Warfare Systems (ASD/RW)
Aeronautical Systems Division
Wright-Patterson Air Force Base, Dayton, Ohio 45433
under Contract F09603-80-G-3338

by
S. Baily
R. Gilbertson
E. Straub

ARINC Research Corporation
a Subsidiary of Aeronautical Radio, Inc.
2551 Riva Road
Annapolis, Maryland 21401
Publication 2302-01-1-2291

ABSTRACT

This document reports on ARINC Research Corporation's work in developing and evaluating alternative acquisition strategies for the USAF's Common Modular Multimode Radar (CMMR) program. It reviews work performed under two contractual efforts: F0603-78-G-4125, which addressed the hardware-related aspects of the program; and F0903-80-G-3338 which addressed the software acquisition and support implications. The work was sponsored by Air Force Systems Command's Deputy for Reconnaissance and Electronic Warfare, Aeronautical Systems Division (ASD/RW). The report reviews the relevance of current software technology and production programs; reviews guidance provided by existing and proposed policies, Directives and Standards; and examines the operational, cost, schedule, risk, supportability, and management aspects of the acquisition alternatives. An acquisition strategy is recommended, and implementation activities are outlined.

FOREWORD

Under contract with Air Force Systems Command's Deputy for Reconnaissance and Electronic Warfare, Aeronautical Systems Division (ASD/RW), ARINC Research Corporation developed and evaluated alternative acquisition strategies for the Common Modular Multimode Radar (CMMR). This document reports on the work and its conclusions.

ARINC Research wishes to acknowledge the excellent cooperation received from the Air Force and Navy representatives who participated in the study. We appreciate particularly the guidance and support provided by the Program Manager, Major Carl Canter; the AFSC/XR JOVIAL Designated Control Agent, Major Al Kopp; and the ASD/ACCX TI-59 Cost Model developer, Capt. Bob Gaffney.

CONTENTS

EXECUTIVE SUMMARY

1.1 SCOPE

This report was prepared by ARINC Research Corporation for the Aeronautical Systems Division's Deputy for Reconnaissance and Electronic Warfare (ASD/RW) under contract F09603-80-G-3338-SC01. It presents the results of a four-month investigation into alternative acquisition strategies for procuring software for the Common Modular Multimode Radar (CMMR) program. In the course of the study—an extension of the work performed under contract F09603-78-C-4125*—we performed a first-order trade-off and risk analysis of the capabilities, performance, and costs associated with various approaches to acquiring CMMR software. The report contains: (1) a review of applicable DoD, USAF, and AFSC guidelines that will affect CMMR software acquisition, (2) an assessment of the development approach and current status of architecture programs for emerging core avionics, (3) identification of other relevant radar software development and test programs, and (4) formulation and evaluation of alternative software acquisition strategies. An underlying assumption during the study was that the CMMR will be procured from the winner of an AFSC-recommended dual fly-off that will pit the Hughes APG-65 F/A-18 radar against an advanced version of the Westinghouse APG-66 F-16 radar; F-16 aircraft will serve as test-beds.

2.1 BACKGROUND

The CMMR program is a software-intensive new radar initiative for FY81. The weapon delivery algorithms, control/display features, stabilization interfaces, electronic counter-countermeasure features, and other very sophisticated capabilities of the system are primarily implemented through software. In addition, the basic design philosophy of the system is "growth through software flexibility." For that reason, the software-acquisition philosophy must be given the same emphasis as that afforded hardware acquisition.

Figure S-1 indicates where and how the key Standards impact the CMMR. Current acquisition guidelines require that the Program Manager adhere to certain Standards unless compelling reasons (cost, schedule, etc.) provide reason for exemption. Key among these Standards are:

- MIL-STD-1553B (21 September 1978); "Aircraft Internal Time Division Command/Response Multiplex Data Bus", which specifies multiplex data bus techniques to be utilized during integration of aircraft avionics subsystems

* See ARINC Research Corporation Publication 1564-11-1-2122, "Development of Acquisition Strategies for the Common Multimode Radar Program"; January 1980.

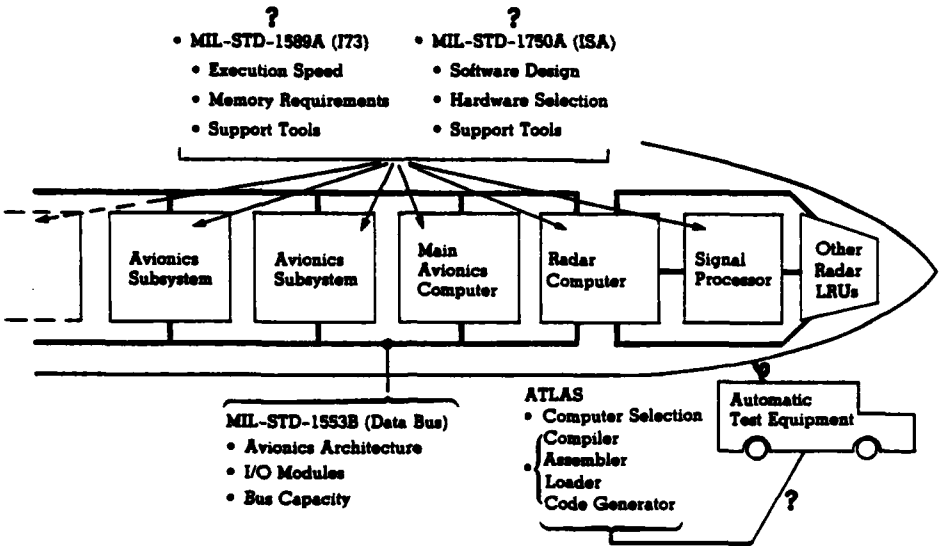


Figure S-1. KEY SOFTWARE STANDARDS AND ISSUES IMPACTING CMMR

- MIL-STD-1750A (2 July 1980); "Airborne Computer Instruction Set Architecture", which defines the instruction set, mnemonics, and data format requirements for airborne computers
- MIL-STD-1589A (15 March 1979); "JOVIAL (J73)", which defines the J73 programming language

3.1 FINDINGS CONCERNING CURRENT SOFTWARE GUIDELINES, TECHNOLOGY, AND PRODUCTION PROGRAMS

The thrust of current guidelines applicable to USAF digital avionics systems' acquisition and development is toward standardization of data interfaces, digital architecture and use of a higher order language (HOL). The major Standards with potential for significant impact on the CMMR program are MIL-STD-1553B (Mux bus), -1750A (ISA), and -1589A (J73 HOL). MIL-STD-1760 (Aircraft Stores/Electrical Interface) is currently in draft form and may be applied to CMMR. Since CMMR is basically a cost-driven standardization initiative and is a highly software-intensive "growth" system, it is reasonable to project that all the Standards will be applied to the program unless compelling reasons for exemption can be shown. Our analysis does not reveal any such compelling reasons. Rather, use of the Standards could provide long-term benefits, even if only partially applied.

A key policy issue is the use of the J73 HOL for the radar operational flight program (OFP). At least partial implementation—with 80 percent of source code instructions as a goal—appears reasonable and desirable. While the timing requirements of portions of the OFP dictate use of assembly language, a HOL can be used for a significant number of the routines subject to frequent change. Since there is a stated "growth" policy with respect to the software for CMMR, this partial HOL approach seems necessary and prudent. While wholly dependable data are not available for an absolute estimate of the LCC savings for a partial HOL approach, it is firmly established that software support costs exceed development costs by as much as 4 to 1 for threat-sensitive systems such as the radar.

A conservative analysis of support savings provided by partial use of the J73 HOL for the F-16 OFP alone indicates that the investment would be amortized within the expected lifetime of the radar system. While this does not present an irrefutable case for the use of a HOL, neither is there a compelling reason to apply for exemption from the current Directives. From a schedule consideration, it appears likely that sufficient development and support tools for J73 will be available for at least one of the competitors by the time of the fly-off.

With respect to the use of MIL-STD-1750 ISA, we perceived an emerging competitive hardware base suitable for the radar computer of CMMR; there are more than 20 models currently under development in the industry. The practicability of implementing a -1750A architecture for the radar programmable signal processor (PSP) is still in contention both in the government and industry, because a very fast-moving technology is involved; further study will be required before a direction for the instruction set architecture (ISA) can be established for this important element of the CMMR architecture.

Software development requires the same developmental process and controls as does hardware development. Air Force Regulation 800-14, Volumes I and II, provides program management guidance for the acquisition and support of computer resources in systems; it should be followed closely.

We reviewed three current production programs to gain insight into precedents and lessons learned about advanced-aircraft radar programs. It is significant that the radar hardware and software designs for all three aircraft examined (F-15, F-16, F/A-18) are still being updated and probably will not mature for several years. Radar memory requirements for all three aircraft have increased substantially due to deficiencies in design and new software requirements. The avionics hardware and software for all three aircraft has been designed for modularity to enhance maintenance.

The F-16 and F/A-18 employ some HOL source coding in their OFPs, but the F/A-18 APG-65 radar OFP is written totally in assembly language and there are no plans to convert to HOL. The F-16 advanced radar OFP is scheduled to be written in J73. Software support for the radar OFP for all three aircraft is accomplished under a prime/subcontractor arrangement. The Air Force as yet has not committed itself to organic support for the software of these aircraft radars, chiefly because there is a lack of knowledge of the software and hardware involved, and a lack of in-house skills to provide the support needed.

Table S-1 presents a summary of key current software acquisition policies, Directives, and Standards reviewed during this investigation and our assessment of their relevance to and impact on system design, choice of hardware, and relative cost. Table S-2 summarizes the development approach, status, and compliance with the key software Standards.

4.1 ACQUISITION STRATEGIES

Three alternative software acquisition strategies were developed and analyzed: responsibility assigned to (1) a prime integrating contractor (PIC), (2) a radar integrating contractor (RIC), or (3) a single operational flight program developer and integrating contractor (SIC). Any one of the three strategies would begin approximately six months before the winner of the dual fly-off between Hughes and Westinghouse is declared. It is assumed

Table S-1. Summary of Documents Reviewed

Document Reviewed	Relevance	Impact			
				Cost	
		System Design	Hardware Choice	Acquisition	Support
Policies					
AFR-800-14	Yes	Yes	No	Unchanged	Unchanged
AFR-65-3,	Yes	Yes	No	Unchanged	Decreased
AFR-300-10	Yes	Yes	Maybe	Increased	Decreased
Directives/Instructions					
5000.29	Yes	Yes	No	Increased	Decreased
5000.31	Yes	Yes	Maybe	Increased	Decreased
5000.5X	Yes	Yes	Maybe	Increased	Decreased
Standards					
MIL-STD-1553B	Yes	Yes	No	Unchanged	Decreased
MIL-STD-1750A	Yes	Yes	Yes	Increased	Decreased
MIL-STD-1589A	Yes	Yes	Maybe	Increased	Decreased
ATLAS	Yes	No	No	Increased	?
Ada	Not Now	No	No	Increased	?

Table S-2. ASSESSMENT OF CURRENT PRODUCTION PROGRAMS

Weapon System	Development Approach	Current Status	Current Compliance with Key Software Standards
F-15	• Prime Integrating Contractor • Centralized Architecture • First digital bus • Extensive use of solid state memories • Assembly language source code	• New radar computer & PSP	N/A: Aircraft developed prior to Standards
F-16	• Prime Integrating Contractor • Decentralized architecture • First 1553 bus • Extensive use of solid state memories • Partial use of a HOL	• New radar under development	Uses: -1553 -1589 HOL(FCC) -ATLAS
F/A-18	• Prime integrating contractor • Distributed architecture • Extensive use of core memories • Partial use of a HOL	• Baseline not yet frozen	Uses: -1553 -CMS-2M HOL (Mission computers)

that the market size will have been established and user requirements firmed-up by this time, and that the market will sustain two or more radar manufacturers.

4.1.1 Prime Integrating Contractor (PIC)

The PIC approach assumes that there would be five individual contractors responsible for the radar OFPs required for the five candidate aircraft. (see Table 2-1). These contractors are assumed to have a better overall knowledge of individual aircraft avionics but would require assistance from the radar manufacturer to integrate the radar hardware and its OFP into the weapon system. The radar could be supplied either as GFE or CFE.

4.1.2 Radar Integrating Contractor (RIC)

The RIC approach assigns responsibility to the radar hardware manufacturer for the hardware production, radar OFP development, and integration of the radar and its OFP into the avionics suites of each aircraft. Assistance from the developer of the avionics main computer hardware and software and the subsystems manufacturers would be required. Because of the quantities of units potentially involved, this approach could well involve two or more radar manufacturers.

4.1.3 Single OFP Developer and Integrating Contractor (SIC)

In the SIC approach, one software contractor develops and integrates all radar OFPs. He would very likely require contractual assistance from both the avionics software and the radar hardware manufacturers. The SIC could team with the radar or avionics developer.

5.1 CONCLUSIONS

The following general conclusions have been reached by ARINC Research.

5.1.1 Review of Software Guidelines, Policies, and Standards

- The thrust of current guidelines applicable to USAF avionics systems' acquisition is toward standardization of data interfaces, use of digital architecture, and use of HOLs. The major Standards with potential for significant impact on the CMMR program are MIL-STD-1553B (Mux Bus), -1750A (ISA), and -1589A (J73 HOL). MIL-STD-1760 (Aircraft Stores/Electrical Interface) is in draft form and may to be applied to CMMR.
- Since CMMR is basically a standardization initiative that is highly software intensive, it is reasonable to project that all the Standards mentioned above will be applied to the program unless compelling reasons for exemption can be shown. Our analysis does not reveal any such compelling reasons. Rather, use of the Standards could provide long-term benefits, even if only partially applied.
- Software development requires the same development process and controls as hardware development. Air Force Regulation 800-14, Volumes I and II, provides program management guidance for the acquisition and support of computer resources in systems; it should be followed closely.

5.1.2 Review of Technology Programs

- Our review of nine radar technology programs for applicability to CMMR software development revealed the following four that will have significant impact on CMMR:

Program Title	PE/Project	Status	Impact on CMMR
RPSP	64201/2519	Continuing	Software algorithms transfer to F-16 if done ASAP
NCTR	63742/1177	Continuing	Algorithms transfer to CMMR
ERIP	64201/2259	Cancelled	TF radar portions applicable to CMMR
EAR	63241/1206	Project Complete	Technology Source for some algorithms and software

- The IDA program establishes and promotes standards for digital avionics architecture but does not enforce their use. Existing facilities available to the Program Manager to provide the capability to implement IDA standards include the ASD/ENASD SEAFAC, the ASD Computer Center, and the AFWAL/AA AVSAIL simulation facility.
- Although the MATE program will not complete FSED until 1985, it could be directed for application to CMMR with use of interim contractor support until MATE hardware is available.

5.1.3 Production Program Review

- The radar hardware and software designs for all three aircraft examined (F-15, F-16, F/A-18) are still being updated.
- Radar memory requirements for all three aircraft have increased by a factor of between 3 and 6 due to deficiencies in design and new software requirements.
- Both the avionics and radar software for all three aircraft have been designed for modularity. The modular software design should enhance software maintenance.
- The F-16 and F/A-18 aircraft employ some HOL source coding in their main computer OFPs. However, the F/A-18 APG-65 radar OFP is written totally in assembly language and there are no plans to convert it to a HOL. The F-16 Advanced Radar OFP is planned to be written in J73.
- Software support for the radar OFP for all three aircraft is accomplished under a prime/subcontractor arrangement at present. The Air Force as yet has not committed itself to organic support.

5.1.4 Acquisition Strategies

- The preferred software acquisition strategy implied by the ranking of our alternatives is a single OFP developer and integrating contractor (SIC).
- The relative importance of the evaluation criteria developed in this report are dependent on Air Force priorities. The apparent thrust of the current government deliberations with respect to CMMR is that, within reason, schedule considerations will defer to cost considerations. The dual competitive fly-off approach appears to be the emerging overall procurement philosophy. Given this philosophy, the assumption that two or more aircraft are involved, and the willingness of the Air Force to accept considerably more technical and management involvement in the program than is normally the case, the following overall acquisition strategy is implied for the total (hardware and software) CMMR program:
  - Use of a single OFP developer for the common software portions of CMMR. If there is a market requirement for more than one radar hardware supplier, the SIC strategy appears to be the preferable approach for the OFP development. Otherwise RIC and SIC appear equally attractive.
  - Single design of the two common major LRUs: Programmable Signal Processor (PSP) and Radar Computer.
  - Use of MIL-STD-1750A ISA and MIL-STD-1553B for at least the Radar Computer. More information is needed on the PSP requirements before the ISA can be established for this LRU.
  - Other possible hardware GFE includes most SRUs of the transmitter, receiver-exciter, antenna and servo, associated controls and displays, and remote terminal units for the multiplex bus installation. The extent to which these units are supplied as GFE is dependent on the target aircraft.
  - A growth-oriented software implementation approach. This implies a structured programming approach following the philosophy, if not the precise requirements, of MIL-STD-1679. Another approach might be to use Ada as a design language to achieve the same objective.
  - Transfer of advanced algorithms, such as NCTR, to facilitate growth from laboratory programs (e.g. PE 63742) at a later time; sponsorship of other algorithms not undertaken by the laboratories to avoid concurrent development expense for each aircraft system.
  - Use of the J73 language for those portions of the architecture that are susceptible to change (e.g. threat-sensitive) and for which execution time can be accommodated. J73 should not be applied indiscriminately for subroutines that are standard in software libraries and that are reasonably stable (e.g. Kalman Filter processing algorithms). However, it is expected that, with time, more and more of these subroutines will be written in a HOL.
  - Use of in-house resources and personnel when possible. For example, SEAFAC could be used for independent verification and validation, if MIL-STD-1750A is directed.
  - Use of GFE compilers for development and support. Initially, this may require the use of GFE machines (e.g. DEC-10) as well. Eventually the Federal Software Exchange Center will provide more portable tools.

- Reliance on the contractor for initial support of both hardware and software to permit more considered examination of the potential long-term alternatives.

5.1.5 Implementation Activities

- The three key actions in the formal direction required from Headquarters USAF are (1) selection of the candidate aircraft, (2) allocation of funds, and (3) issuance of the PMD.
- The formal direction from Headquarters USAF should be reflected in the ASD Program Management Plan (PMP).
- The most critical initial step following the submittal of the PMP is a definitive operational requirements audit, followed by (1) formation of a Test and Evaluation Working Group (TEWG), (2) formation of an Interface Control Working Group (ICWG), and (3) conduct of a series of LCC tradeoffs.

6.1 RECOMMENDATIONS

ARINC Research recommends that Air Force planners responsible for the CMMR effort take the following actions:

- Use a single OFP developer and integrating contractor for the CMMR software acquisition strategy.
- Develop a detailed activity road map to implement the acquisition strategy.

CHAPTER ONE

INTRODUCTION

1.1 SCOPE

This report was prepared by ARINC Research Corporation for the Aeronautical Systems Division's Deputy for Reconnaissance and Electronic Warfare (ASD/RW) under contract F09603-80-G-3338-SC01. It presents the results of a four-month investigation into alternative acquisition strategies for procuring software for the Common Modular Multimode Radar (CMMR) program. In the course of the study—an extension of the work performed under contract F09603-78-C-4125*—we performed a first-order trade-off and risk analysis of the capabilities, performance, and costs associated with various approaches to acquiring CMMR software. The report contains: (1) a review of applicable DoD, USAF, and AFSC guidelines that will affect CMMR software acquisition, (2) an assessment of the development approach and current status of architecture programs for emerging core avionics, (3) identification of other relevant radar software development and test programs, and (4) formulation and evaluation of alternative software acquisition strategies. An underlying assumption during the study was that the CMMR will be procured from the winner of an AFSC-recommended dual fly-off that will pit the Hughes APG-65 F/A-18 radar against an advanced version of the Westinghouse APG-66 F-16 radar; F-16 aircraft will serve as test-beds.

1.2 BACKGROUND

Program element 64412F, Common Modular Multimode Radar (CMMR), is a new Air Force program initiative for FY81. The program is intended to develop an attack radar capability for multiple aircraft weapon systems through the application of common (core) radar hardware in conjunction with a computer and programmable signal processor. The processor will provide the ability to select modes of operation tailored to the individual role and mission of each aircraft type.

The overall Air Force objective of the CMMR engineering development program is to fabricate and test a modified production multimode radar system that will satisfy the needs of several users in air-to-air and air-to-surface missions. The system must satisfy autonomous all-weather requirements of post-1984 strategic and tactical aircraft through a flexible architectural concept involving the radar's hardware and software capabilities. At the present time, the Air Force expects the CMMR to satisfy its all-weather tactical and strategic mission requirements for such diverse aircraft as the F-16, F-4, F-106, B-1, B-52, and FB/F-111.

---

* See ARINC Research Corporation Publication 1564-11-1-2122, "Development of Acquisition Strategies for the Common Multimode Radar Program"; January 1980.

Air-to-ground modes under consideration include ground search, detect, and track capabilities that would increase existing fire-control and weapons-guidance effectiveness against moving and stationary targets during night operation and under all weather conditions for a variety of missions. This development is to include a terrain-following mode to permit aircraft operation at very low altitudes. Air-to-air modes under consideration would be compatible with current and advanced missiles, permitting multiple kills per intercept.

These planned improvements are to be implemented with improved CMMR antenna radiation characteristics and advanced digital processing of radar data to make the aircraft less detectable and less susceptible to current and projected electronic countermeasures. The overall concept is for new and different threats to be countered by software rather than hardware changes, once the hardware design is frozen.

Under our previous contract, we examined three hardware acquisition strategies—sole source, full competition, and dual source. Each of these strategies displayed certain advantages and disadvantages, but the particular set of schedule and budget circumstances associated with CMMR, favored the dual source strategy, wherein two candidate radars would be developed and a fly-off used to select a winner. The major disadvantage of this strategy is that it does not yield at the outset a "full up" CMMR but rather one with potential to achieve the advanced capability later on.

It was evident in our previous work that a significant portion of the production cost of the CMMR will be attributed to its software-intensive radar computer and programmable signal processor. It was especially significant that the CMMR is to be designed as a "growth" system, with added capabilities being provided through new software additions or modifications to existing software. It is important, therefore, that the cost and schedule effects of different software acquisition strategies be considered with the same emphasis as those of hardware acquisition strategies. We recommended that software acquisition strategies be examined early in the program in order to define alternatives that will facilitate the support and maintenance of evolving software. This strategy should encompass the Standards to be used (1553, 1750, 1589, ATLAS, etc.) and the software support tools. Figure 1-1 indicates where and how the key Standards impact the CMMR.

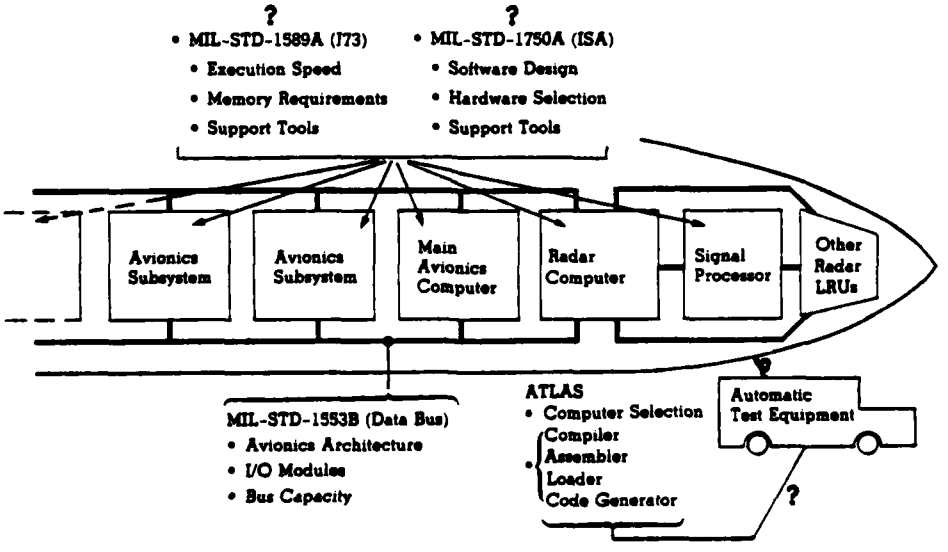


Figure 1-1. KEY SOFTWARE STANDARDS AND ISSUES IMPACTING CMMR

1.3 TASK DEFINITION AND APPROACH

The statement of work for this effort tasks ARINC Research to perform a study and analysis of alternative software acquisition strategies for use by the Aeronautical Systems Division in the CMMR program. The following three tasks were specified:

"Task 1—Review DOD and USAF Avionics Software Guidelines, Policies, and Standards—Review all current avionics software acquisition policies, directives, and standards for possible applicability to the CMMR Program. As a minimum, evaluate the status of current DoD Higher Order Languages (i.e., J73, Ada, etc.) and Standards (i.e., MIL-STD-1750 and 1553) for relevance and impact on CMMR design, hardware, and costs."

Our approach to this task was to review documentation of previous Air Force work and interview appropriate DoD personnel to determine current software acquisition policies, directives, and Standards for possible applicability to the CMMR program. Our review was structured to include the following areas:

- Previous contractor and DoD efforts
  - TRW, LOGICON, Hughes, SCI, etc.
  - Joint Logistics Commanders' Initiatives
  - JOVIAL and MIL-STD-1750 Users Group work
- Key existing documents/Directives
  - DoD Directives and Instructions
  - Military Standards
  - Air Force Regulations
  - AFSC policy and direction

In addition, we interviewed key software personnel at DoD, USAF and AFSC Headquarters, ASD, NAVAIR SYSCOM, and NAVSEA SYSCOM.

"Task 2—Assess Status of Present Avionics Software Technology and Production Programs —Assess the development approach and current status of emerging core avionics architecture programs (such as the Integrated Digital Avionics (IDA) program and Modular Automatic Test Equipment (MATE) program), key digital avionics and support programs, and the present F-15, F-16, and F-18 radar software development/test programs. The progress of the Air Force and Navy software standardization and test approaches in complying with Task 1 directives and guidelines shall be assessed."

Our approach to this task was to acquire and compare current documentation on:

- Core avionics programs
- Avionics architectures, software and radar hardware of the F-15, F-16, and F/A-18 aircraft
- Key Program Management Directives

In addition, key software program personnel of the System Project Offices (SPOs) at ASD and NAVAIR were interviewed. Field trips were made to the Ogden and Warner Robins Air Logistics Centers.

"Task 3—Formulate Software Acquisition Strategies —Based on the results of Tasks 1 and 2, as well as the previously developed hardware acquisition strategies, study and analyze alternative CMMR software acquisition strategies as identified or approved by the ASD Program Office, evaluate CMMR impact of each strategy in regard to supportability, operational capability, cost, schedule, and risk. Based on these analyses, a program matrix of alternative strategies with assumptions, factors, and criteria shall be outlined in such a manner as to permit the Air Force to define an optimum CMMR software acquisition program for pertinent program conditions."

Our approach to this task was similar to that used in our previous development of hardware acquisition strategies. We analyzed the impact of alternative CMMR software acquisition strategies on supportability, operational capability, cost, schedule, and risk. A program matrix was developed to highlight these factors in relation to specific assumptions so that the Air Force could use these data to assist in defining an optimum CMMR software acquisition program for pertinent program conditions.

1.4 REPORT ORGANIZATION

Chapter Two summarizes the current avionics software acquisition policies, Directives, and Standards and their potential impact on the CMMR program.

Chapter Three presents a synopsis of the development approach and current status of emerging core avionics architecture programs, key digital avionics and support programs, and the present F-15, F-16, and F/A-18 radar software development and test programs. It establishes the basis for the approaches and baseline data used in Chapter Four.

Chapter Four presents an analysis of several alternative CMMR software acquisition strategies.

Chapter Five discusses implementation activities.

Chapter Six provides a summary of our findings, draws conclusions, and presents our recommendations.

CHAPTER TWO

REVIEW OF DOD AND USAF SOFTWARE GUIDELINES, POLICIES, AND STANDARDS (TASK 1)

2.1 INTRODUCTION

In the past, the Air Force's acquisition efforts for avionics and support software have been characterized primarily by a proliferation of programming languages associated with contractor-unique machines. The impacts of this on the Air Force have been skyrocketing costs, dependence on weapon system/subsystem developers for individual software support, and a general loss of control over the computer aspects of total weapon system operation and support. Recent initiatives within DoD, however, are pointing the way toward increasing standardization of military computers. Standards now exist for higher-order languages (HOLs), computer architectures, and time-division multiplex data bus techniques. The purpose of this chapter is to examine (1) the current avionics software acquisition policies, Directives, and Standards for possible applicability to the CMMR program and (2) the Air Force's software management guidance appropriate to the CMMR program.

2.2 APPROACH

The review of DoD and Air Force guidelines, policies, and Standards for avionics software, as required under Task 1, was conducted in three areas: previous studies, existing documents and Directives, and interviews with software policy and control agencies. A listing of key documentation appears in Appendices A and E. The key individuals interviewed are listed in Appendix B.

After completing this work, we reviewed the Air Force's management guidance techniques for software acquisition to define the pertinent steps required of the CMMR Program Manager. A starting place for this review was a series of management guides published in 1976 by LOGICON under an Air Force contract. These four volumes, although four years old, remain valid handbooks for use by a Program Manager responsible for software acquisition. Beginning in 1977, a series of "Airborne Systems Software Acquisition Engineering Guidebooks" published by TRW, also under an Air Force contract, have outlined applicable Regulations, Specifications, Standards, etc., in this area. A more recent (October 1979) source of information was the report of the Software Management Subgroup of the Joint Logistics Commanders. This document included guidance regarding software acquisition policy, as well as acquisition, documentation, and quality assurance standards.

Our review of existing directives that will impact CMMR software acquisition included pertinent DoD Directives and Instructions, Military Standards, and Air Force Regulations.

Key software personnel within DoD, HQ USAF, HQ AFSC, ASD, and NAVAIR were also interviewed.

Our results are summarized in the two remaining sections of this chapter: Embedded Computer System Standards and Software Acquisition Management.

2.3 EMBEDDED COMPUTER SYSTEM STANDARDS

Three military Standards for embedded computer systems form the basis for the thrust of near-term Air Force policy concerning software acquisition:

1. MIL-STD-1553B (21 Sep 78): "Aircraft Internal Time Division Command/Response Multiplex Data Bus"
2. MIL-STD-1750A (2 July 80): "Airborne Computer Instruction Set Architecture"
3. MIL-STD-1589A (15 Mar 79): "JOVIAL (J73)"

A fourth standard, MIL-STD-1760 (May 1980), "Aircraft/Stores Electrical Interface Definition", is currently in draft form. All four Standards are contained in the latest draft CMMR directive.* We have reviewed their applicability to CMMR under three categories: data interface standardization, instruction set architecture (ISA) standardization, and HOL, standardization.

2.3.1 Standardization of Data Interface

DoD data interface requirements for aircraft are defined in MIL-STD-1553B and -1760. MIL-STD-1553B specifies requirements for aircraft internal time division command/response multiplex data bus techniques to be used during integration of aircraft avionics subsystems. This Standard replaced the "A" version in September 1978. The primary changes made to -1553A were those required to make the Standard more universally acceptable to other than Air Force users. The changes in the "B" version included definition of mode commands, status bits, addition of a broadcast capability, better coupling definition, and an improved noise test. This tri-service Standard is now widely accepted by the services and contractors.

2.3.1.1 MIL-STD-1553B and CMMR

Most of the current configurations of candidate CMMR aircraft have or will have data busses. Only the F-16 has a 1553 bus at this time. A decision on whether to employ MIL-STD-1553B in the other candidates will be required. Table 2-1 shows the current configurations of the CMMR candidate aircraft relative to the use of the 1553 Mux Bus, avionics HOLs, and the Abbreviated Test Language for All Systems (ATLAS) automatic test equipment HOL.

MIL-STD-1760, "Aircraft/Stores Electrical Interface Definition," specifies the electrical characteristics required to interface expendable stores with the aircraft. This Standard is now in draft form. For the CMMR program it must be compatible both in electrical characteristics and data format with MIL-STD-1553B. Even though no interface difficulty

*HQ USAF Draft PE 64201/2519 PMD, Updated (Circa Apr 80)

Table 2-1. CONFIGURATIONS OF CMMR CANDIDATE AIRCRAFT

	1553 Mux Bus	Avionics Computer	Avionics HOL	ATE HOL
F-16	Yes	M362F-2	J3B-2	Planned
FB/F-111A	Planned	CP2	J73 Planned	No
F-111D, F	Planned	CP2EX*	J73 Planned	No
F-111E	No	Analog	No	No
B52G/H (OAS)	Yes	AP101C	J3B-2	No
F4E/G	No	No	No	No
F-106	No	No	No	No

* Computer will be reconfigurable to MIL-STD-1750A

is envisioned at this time, the CMMR Program Manager must keep aware of MIL-STD-1760 status and pending usage, in case his program is extended to other aircraft in which use of this Standard is planned.

2.3.2 Standardization of Instruction Set Architecture (ISA)

There is a wide variety of instruction set formats in the computer industry. As mentioned in the Introduction, the proliferation of unique computer hardware elements, each with its own instruction-set repertoire, has created a situation in which DoD managers have inadequate control over the computer hardware (and to a certain extent, the software) embedded in the various weapon systems. In November 1978, the Office of the Secretary of Defense for Research and Engineering, in an effort to halt this proliferation, directed that no new computer architectures be developed. As a result, DoD Instruction 5000.5X will be published shortly, specifying the approved list of DoD ISAs to be used during computer hardware development. The list will include the Air Force's MIL-STD-1750A.

2.3.2.1 Air Force Policy Concerning ISA Use

The Air Force policy concerning ISA use was explained in an AFSC/CC message dated 1 March, 1980. The policy is to use MIL-STD-1750A for all aircraft programs that develop or procure embedded computer systems. Requests for waivers to this policy will be examined on a case-by-case basis. A MIL-STD-1750 Users Group, consisting of representatives from the services and industry, was formed in August 1979. It meets periodically to discuss and resolve issues relating to use of the Standard and to take whatever actions are necessary to keep it current. As a result of its concentrated efforts, MIL-STD-1750, dated 21 February, 1979, was updated to -1750A on 2 July, 1980. The CMMR Program Manager should call upon this users group as needed if MIL-STD-1750A is used in the program.

2.3.2.2 MIL-STD-1750A and CMMR

The application of MIL-STD-1750A to the CMMR program has advantages and disadvantages. On the positive side, the -1750A Standard ISA will allow for easier structuring of software. In addition, the Air Force will not be tied to a specific piece of hardware. It

will be able to use any hardware that meets the Standard, thereby providing vendor independence and avoiding "freezing" the technology of any computer design. Specifying the ISA will also allow the Air Force to exploit a common base of software-support resources, including the hardware, software, and personnel required for any particular program. Additionally, ISA definition could allow for the early initiation of software development prior to hardware delivery.

There are also some negative aspects of applying MIL-STD-1750A to the CMMR program. The Standard is not now used in either of the candidate radars; hardware and software would need to be developed to integrate -1750A hardware and software into existing production systems. Since -1750A is relatively new, there is very limited computer hardware available, which circumstance could cause an adverse scheduling impact on the CMMR program. (It is our understanding than there are more than 20 manufacturers building -1750A machinery, however). Software development costs will probably be higher, since there is little practical experience in this area and existing software flight programs would need recoding. Additionally, a whole set of new software support tools—both automatic test equipment (ATE) hardware and software—must be developed, which will undoubtedly add to development costs.

2.3.3 Standardization of Higher Order Language (HOL)

HOLs provide easier man-to-computer understanding than assembly languages by reducing the gap between problem-solving formulation and actual instruction coding. For some time, DoD has recognized the advantages of using HOLs for programming and coding in terms of common compilers, code generators, link loaders, and other common software support tools, once the language has been fully developed. The problem with standardizing HOLs is threefold: (1) the need to develop languages that are most widely applicable, (2) enforcement of their use, and (3) control of their configuration.

DoD policy regarding management of computer resources is established in DoD Directive 5000.29, dated 26 April, 1976. This Directive states, "DoD approved High (SIC) Order Programming Languages (HOLs) will be used to develop Defense system software unless it is demonstrated that none of the approved HOLs are cost effective or technically practical over the system life cycle". In the past four years since this Directive was promulgated, advances in technology have further reduced the "cost" of computer hardware in terms of size, power, and memory. This hardware revolution has greatly reduced the advantages that assembly or machine language might have had over a more "memory consuming" HOL. Therefore, the only reason to request a waiver to the HOL requirement would probably be for technical feasibility, with an emphasis on response time considerations (such as the demanding speed requirements of certain executive and air-to-air radar mode applications) or for meeting specific memory requirements.

DoD Instruction 5000.31, dated 24 November 1976, specifies the following six DoD standard HOLs that will be used (controlling agency shown in parentheses):

COBOL	(DoD)	CMS-2	(Navy)
FORTRAN	(DoD)	SPL-1	(Navy)
JOVIAL (J3B and J73)	(Air Force)	TACPOL	(Army)

An unpublished revision to 5000.31, dated 3 April, 1980, identifies a new HOL listing:

FORTRAN	(Air Force)	CMS-2M	(Navy)
JOVIAL J73	(Air Force)	Ada	(Joint Program Office)
CMS-2Y	(Navy)	ATLAS	(IEEE)

2.3.3.1 Air Force Policy Concerning HOL Use

The Air Force has stated that J73 (MIL-STD-1589A) will be used for avionics real-time applications and ATLAS will be used for test applications.

In its function as the Designated Control Agent for J73, AFSC/XR stated in a message dated 15 May, 1980, that "DoD Instruction 5000.31 specifies a number of standardized HOLs to be used in the development/acquisition of embedded computer system software programs. Of these HOLs, it is Air Force Systems Command policy to use J73 as the preferred interim Air Force standard HOL until Ada is available." This message further identified 15 development programs as candidates now for J73. These programs are listed in Table 2-2 along with six DoD programs that are committed to J73. CMMR does not appear on this list, probably because it is a new start for FY81 and a CMMR Program Management Directive (PMD) has yet to be issued.

Ada, which will be the new DoD Standard, will not be fully available until the mid 1980s. Because of this, the Air Force intends to phase out the J73 language over a two-year period beginning in 1984. It should be noted that an updated version of MIL-STD-1589A (-1589B) has been approved by HQ AFSC. It should be released shortly, and DoD Instruction 5000.31 is planned to be revised accordingly.

Table 2-2. CANDIDATE PROGRAMS FOR J73 APPLICATION

AFSC Programs as of 15 May, 1980

SPACE DIVISION	ELECTRONIC SYSTEMS DIVISION
Defense Meteorological Satellite	SEEK TALK
Space Defense System	RAPS & CAFMS
Interface/Converter Computer Project	OASIS
Ground Control Network III	ATEC
ARMAMENT DIVISION	**AERONAUTICAL SYSTEMS DIVISION**
Ground Jammer	Precision Location Strike System
WASP	USIP/IFFN
Modular Threat Emitter	Fusion
DTS	

DOD Programs

MX	Pershing
DIS	CHOL Development
DAIS	AMRAAM

2.3.3.2 Status of MIL-STD-1589A

A JOVIAL Users Group, consisting of representatives from the services and industry, meets periodically to consider issues relative to the JOVIAL language. Various committees, such as JOVIAL Experience, User Requirements, Compiler Issues, and Support Tools, propose and review changes and submit their recommendations to the Users Group. A Language Control Board reviews all recommendations and advises the Language Control Agent on specific issues. The CMMR Program Manager should call upon this Group as needed, if MIL-STD-1589A is used in the program.

To date, several J73 compilers have been developed by the Air Force. Table 2-3 lists the status of the J73/1750 support software tools hosted on both the IBM-370 and DEC-10 computers, either available now or under development.

Table 2-3. AVAILABILITY OF J73 SUPPORT SOFTWARE TOOLS

TOOL	DATE AVAILABLE
MIL-STD-1750	
Compiler	Now
Code Generator	Now
Assembler	Now
Linker	Now
Simulator	Now
MIL-STD-1750A	
Compiler	15 Dec 80
Code Generator	15 Dec 80
Assembler	15 Feb 81
Linker	15 Feb 81
Simulator	15 Feb 81

McDonnell Douglas Aircraft Corporation (MCAIR) is also developing a MIL-STD-1750A support software package for ASD/XR, which will be hosted on VAX 11/780, PDP 11/60, and ITEL AS/5 (IBM-370 compatible) computers. The maintenance handbook should be delivered in December 1980; documentation for the Assembler, Linker, and Simulator is expected to be delivered in October 1980.

The Air Force's Armament Laboratory is also developing a J73 compiler for the Z-8000 to be hosted on the IBM-370. It is scheduled for delivery by December 1980. The Air Force will also develop a J73 compiler hosted on the VAX 11/780 and targeted for the Westinghouse MILLI-EP avionics radar computer for the F-16 advanced radar discussed in the next chapter. ASD J73 language and compiler development courses are planned for October and December 1980.

2.3.3.3 MIL-STD-1589A and CMMR

As with MIL-STD-1750A, there are advantages and disadvantages to applying the J73 HOL to the CMMR program. Obviously, a standard HOL that is used for many systems by

industry and the DoD would give the Air Force greater flexibility in its software maintenance concept. As J73 becomes more widely used, a large support pool of experienced personnel and tools should eventually become available. Programs written in a HOL are easier to follow and update. By using a HOL, the Air Force would not require software support from the developer or integrator for the entire lifetime of the system. Source code is cleaner, patches are better written, and problems can be located faster. Unfortunately, 100 percent HOL usage for such efforts as a radar's Operational Flight Program (OFP) probably is not feasible because of processing speed and memory constraints. Also, neither the APG-65 nor the APG-66 candidate radar currently has flight programs written in this HOL. Although there are J73 compilers and support tools available, some have not been fully validated by the Air Force. This could result in a delay for the CMMR program. Producing compilers and other required software support tools could increase the development cost associated with HOL use above that required for assembly language. Corrective maintenance of avionics software is typically done in assembly language even for OFPs coded originally in a HOL.

CMMR is envisioned to be a "growth" system, which portends many software changes for many aircraft. It appears appropriate, then, to employ software that is easily understood and modified.

2.3.4 Software Development Standards

There is no Air Force accepted software development Standard, although a Rome Air Development Center (RADC) specification does exist. The only DoD Standard currently in existence is MIL-STD-1679 dated 1 December, 1978. This was developed by the Navy to prescribe how its computer programs should be constructed during software development. The primary purpose of MIL-STD-1679 is to insure that embedded computer software is designed to facilitate efficient update over the operational life of the system. Standardized hierarchial structures that partition the software into realizable components identifiable with functional requirements are defined. Detailed documentation describing the software is required so that changes may be made by personnel not associated with the original development. The advantages that this form of structured programming provides to the CMMR program are supportability (which is essential to a growth oriented system) and potential life cycle cost (LCC) saving. Disadvantages include possible increases in development cost as well as the constraints imposed on the developing contractor.

2.3.4.1 Software Development Standards and CMMR

The issue of tri-service software development standards is currently being examined by a software working group of the Joint Logistics Commanders (JLC). Early results of this work could be useful to the CMMR Program Manager during CMMR acquisition. We do not feel, however, that this is a critical issue for the program at this time. Most software developers in the industry are employing some form of structured programming as part of their in-house procedures. While these may not be as stringent as those published in a Military Standard, they will provide many of the essential benefits.

2.4 SOFTWARE ACQUISITION MANAGEMENT

The JLC Computer Resources Management Group is studying the services' software acquisition policies, procedures, and standardization efforts. The goal of these efforts is to formulate and recommend actions to the JLC or DoD agencies that will improve the software acquisition process. These on-going initiatives in computer-resources acquisition

policy are expected to result in (1) an overall narrowing of a Program Manager's software options, (2) new Standards pertaining to interfaces and peripheral hardware, (3) growth in available support software tools and experienced personnel, and (4) significant use of the Federal Software Exchange Center (a GSA-managed repository for computer programs similar to the National Technical Information Service's repository for documents).

Table 2-4 presents a summary of current software guidelines, policies, and Standards and their potential impacts on the CMMR program. Most significant is AFR-800-14. It provides Program Managers with guidance for acquisition, management, and support of computer resources. Volume I, dated 12 September, 1975, outlines Air Force policy relative to computer resources; it implements DoD Directive 5000.29. Volume II, "Acquisition and Support of Computer Resources in Systems," dated 26 September, 1975, is a Program Manager's "handbook" for computer resource acquisition management.

Table 2-4. SUMMARY OF SOFTWARE GUIDELINES, POLICIES, AND STANDARDS

Document	Intent	Impact on Program	When to Address	Implication on Radar		Issues	Resolution
				APG-65	APG-66		
DOD D 5000.29 AFR-800-14 DOD I 5000.31 AFR-300-10	Embedded Computer System management and control	Must address software management and development in PMP; requires HOL	Now	N/C*	N/C	None	Form CRWG; write CRISP; develop CPDP
MIL-STD-1589A	Defines J73	Directed to use	Now	N/C	N/C	Not used now; compiler availability; reduced speed; increased cost	Make partial use of HOL
DOD 5000.5x MIL-STD-1750A	Defines ISA	Directed to use	Now	N/C	N/C	Not used now; hardware not available	Identify computers
MIL-STD-1521A	Technical reviews and audits	Must establish DID/CDRL items	1981	OK	OK	Documentation status; No standard set of acceptance criteria	Under analysis by JLC
MIL-STD-480 -483 -490 AFR-65-3 AFSCP-800-7	Configuration management and control; CPCI Format	Requires formal control of software configuration	1981	OK	OK	ICD status for candidates unknown	Form ICWG; develop CMP
MIL-S-52779	Software QA	Dictates contractor rqmts for QA pgm	1981	OK	OK	No DID available; difficult to enforce	Under Analysis by JLC
MIL-STD-1553B	Defines Mux bus	Directed to use	1982	N/C (1553A)	N/C (1553A)	Not all candidate aircraft have bus	Use additional interface modules
MIL-STD-1760	Defines aircraft stores interface	Directed to use	When finalized	?	?	In draft form; interoperability of 1760/1553B	Use additional interface modules
MIL-STD-1679	Defines software development	Requires structured programming, etc.	If adopted by USAF	N/C	N/C	USAF does not have an equivalent	Under analysis by JLC

* Non-compliant

It is important for the CMMR Program Manager to realize that software development requires the same process and controls as hardware development. Software designs should be frozen before coding starts. The requirements of AFR-800-14 applicable to the CMMR Program Manager are summarized by program phase in the following subsections.

2.4.1 Planning Phase

The key management planning tools addressed by AFR 800-14 are the Program Management Plan (PMP), the Computer Resources Working Group (CRWG), the Computer Resources Integrated Support Plan (CRISP), and the Computer Program Development Plan (CPDP). The PMP is developed by the Program Manager and presents a time-phased plan for activities and resources required to accomplish the activities outlined in the PMD. The PMP must address software resources as well as hardware resources. The CRWG identifies key players required to direct the development of the system's computer resources. It must include representation from the developing, supporting, and using commands, as well as from the Air Force Test and Evaluation Center and the Air Training Command when appropriate. The CRWG is responsible for preparing and updating the CRISP. It identifies organizational relationships and responsibilities for the management and technical support of computer resources throughout the system life-cycle. The CPDP is also the responsibility of the implementing command; it identifies the actions required to develop and deliver computer program configuration items and necessary support resources. The Program Manager normally directs the CPDP as a deliverable to be approved under the contract.

2.4.2 Engineering Development Phase

During the engineering development phase, reviews and audits are required on software products just as for hardware. MIL-STD-1521A, "Technical Reviews and Audits for Systems, Equipment, and Computer Programs" provides direction for the conduct of these reviews and audits.

2.4.3 Testing Phase

Computer program testing parallels hardware testing. A comprehensive test plan must be developed and tests should be conducted at the software module level, hardware/software interface level, and operational system level. Independent verification and validation is needed during all phases of software development. It must be performed by an organization that did not develop the code. It is a tool that can help to insure an orderly development process through early identification and correction of errors.

2.4.5 Configuration Management

Configuration management is important throughout the system life cycle; it is directed by Air Force Regulation 65-3. MIL-STD-480 (Configuration Control, Engineering Changes and Waivers), -483 (Configuration Management Practices for Systems, Equipment Monitoring and Computer Programs), and -490 (Specification Practices) must be followed. Interface Control Working Groups (ICWGs) are necessary and software Interface Control Drawings (ICDs) must be developed and maintained.

2.5 SUMMARY

Standardization of computer resources has been a major policy initiative within DoD for many years. DoD Instruction 5000.31 specifies the use of six standard HOLs to prevent continued HOL proliferation. Air force guidance has further reduced this to one (J73) for real time avionics applications. Continued ISA proliferation will be prevented by the proposed DoD standard 5000.5X and, in the Air Force, by MIL-STD-1750A. The overall goal of these standardization initiatives is reduction of life-cycle cost.

The CMMR Program Manager undoubtedly will be required to comply with some standards for embedded computer systems. These standards could include MIL-STD-1553B for a multiplex data bus, MIL-STD-1589A (B) for the HOL (J73), and MIL-STD-1750A for the ISA. The combination of the J73 HOL and 1750A ISA Standards implies a unique set of development and support tools. It is anticipated that the development and use of these tools for the CMMR will increase the "front end" cost of the program. However, the implementation of these Standards could also be expected to reduce the life cycle cost of a growth-oriented system such as the CMMR. The use of MIL-STD-1679 for software development, or one similar, could also result in software reliability and maintainability cost savings. The ability to change support contractors and the anticipated relatively fast implementation of software changes made possible by the use of these Standards should outweigh the initial cost increase if the equipment is maintained in the inventory for a sufficiently long period. This subject is examined quantitatively in Chapter Four.

Table 2-5 presents a summary of current software acquisition policies, Directives, and Standards reviewed during this investigation and our assessment of their relevance to and impact on system design, choice of hardware, and cost.

Table 2-5. Summary of Documents Reviewed

Document Reviewed	Relevance	Impact			
				Cost	
		System Design	Hardware Choice	Acquisition	Support
Policies					
AFR-800-14	Yes	Yes	No	Unchanged	Unchanged
AFR-65-3	Yes	Yes	No	Unchanged	Decreased
AFR-300-10	Yes	Yes	Maybe	Increased	Decreased
Directives/Instructions					
5000.29	Yes	Yes	No	Increased	Decreased
5000.31	Yes	Yes	Maybe	Increased	Decreased
5000.5X	Yes	Yes	Maybe	Increased	Decreased
Standards					
MIL-STD-1553B	Yes	Yes	No	Unchanged	Decreased
MIL-STD-1750A	Yes	Yes	Yes	Increased	Decreased
MIL-STD-1589A	Yes	Yes	Maybe	Increased	Decreased
ATLAS	Yes	No	No	Increased	?
Ada	Not Now	No	No	Increased	?

CHAPTER THREE

ASSESSMENT OF CURRENT AVIONICS SOFTWARE TECHNOLOGY AND PRODUCTION PROGRAMS (TASK 2)

3.1 INTRODUCTION

The purpose of our work under Task 2 was to gain additional insight into the Air Force's current avionics software technology through a detailed review of ongoing 6.3 and 6.4 programs and by analysis of DoD's three most advanced digital avionics multirole aircraft. The work provided the basis for the alternative software acquisition strategies formulated under Task 3.

The information contained in this chapter was obtained from publications and interviews with industrial and service-connected personnel. A listing of the principal source material appears in Appendices A and D; the key individuals interviewed are listed in Appendix B.

This chapter updates our report on the status of the Air Force radar software technology efforts presented in ARINC Research publication 1564-11-1-2122 submitted in January 1980 (see Section 1.1). It also discusses two major Air Force avionics and test equipment initiatives: the Integrated Digital Avionics (IDA) and Modular Automatic Test Equipment (MATE) programs. A review of the Air Force and Navy approaches to managing avionics programs is included to highlight differences and, where possible, indicate trends for the CMMR Program Manager. Finally, we analyzed the existing and planned avionics architecture and subsystems of the F-15, F-16, and F/A-18 weapons systems, concentrating on their central computers, avionics software, and radar software and hardware.

3.2 TECHNOLOGY REVIEW AND UPDATE

The purpose of this section is to present an update of the status of radar technology efforts that have undergone significant change since our report of January 1980. We have added reviews of the IDA and MATE programs.

3.2.1 Current Status of Radar Technology Programs

Table 3-1 presents a summary of the current status of the programs examined in our January, 1980 report. Details are provided in the following subsections.

3.2.1.1 Radar Programmable Signal Processor (PSP)—64201/2519

Phase I, Task I of the PSP Radar Improvement Program (formerly called the FIST program) contains several new software modes to be demonstrated with the F-15's new PSP

Table 3-1. TECHNOLOGY PROGRAMS REVIEWED

Title	Program Element/Project	Status	Potential Impact on CMMR
Radar Programmable Signal Processor (RPSP)	64201/2519	Continuing	Software algorithms transferable to F-16 if done ASAP
Non-Cooperative Target Recognition (NCTR)	63742/1177	Continuing	Algorithms transferable to CMMR
ECCM Radar Improvement (ERIP)	64201/2259	Cancelled	Terrain following radar portions applicable to CMMR
Advanced Strike Radar Technology (ASRAT)	63203/69DF	Near-term program cancelled	Results of long-term program will be too late for CMMR fly-off
Advanced Fighter Technology Integration (AFTI)	63205/2506 63245/2061	Continuing	Little applicability now; later potential
Low Altitude Navigation Targeting Infrared Night (LANTIRN)	63249/2693	Production hardware availability slipped 1 year from previous report	No impact forseen at this time; could affect terrain-avoidance requirement for flight tests at later time.
Assault Breaker (PAVE MOVER)	63747/2217	Continuing	Little Applicability
Advanced Medium Range Air-to-Air Missile (AMRAAM)	63370/2437	Continuing	MIL-STD-1750, 1553, and 1589 apply to AMRAAM; -1760 may also be directed if approved; CMMR must be compatible with AMRAAM interfaces
Electronically Agile Radar (EAR)	63241/1206	Project Complete	Technology source for some algorithms and software

(MX-10064/APG): passive ranging, angle-on-jam, long range track while scan (TWS), and short and medium range TWS. Additionally, a software streamlining task, improved displays (PSG/SPS), and elimination of blind zones (3/8 Detection) are also to be demonstrated during this phase.

System design reviews (SDRs) for all modes except 3/8 Detection and PSG/SPS are complete; these are scheduled for SDRs by October 1980 and February 1981, respectively. Following the SDR of each mode, the software will be integrated into the F-15 test-bed aircraft and flight tested. Current plans call for the forwarding of software documentation to the F-16 SPO after all flight testing has been completed (April 1982). After our initial CMMR work, we recommended that each software algorithm be transferred to the F-16 SPO *immediately after the SDR* so that all modes might be integrated with the candidate CMMRs in the two F-16 test-bed aircraft prior to competitive fly-off. This recommendation can still be followed if prompt action is taken.

3.2.1.2 Non-Cooperative Target Recognition (NCTR)—63742/1177

The results of the effort to develop and integrate identification algorithms into the F-15 PSP are directly applicable to the CMMR, Advanced Medium Range Air-to-Air

Missile (AMRAAM), and identification-friend-or-foe (IFF) programs. The program is on schedule, and, since the CMMR program has slipped by six months, the NCTR program should now be able to meet the CMMR schedule without the acceleration recommended in our earlier report.

3.2.1.3 ECCM Radar Improvement Program (ERIP)—64201/2259

The F-111 radar modifications and flight tests that were scheduled under the ERIP program are in abeyance awaiting a decision on whether CMMR will be incorporated into the F-111. The results of this work would be directly applicable to the CMMR terrain following/terrain avoidance (TF/TA) modes.*

3.2.1.4 Advanced Strike Radar Technology (ASRAT)—63203/69DF

A Congressional funding cut has caused the near-term phase of the ASRAT program to be cancelled. The near-term rf cueing effort will continue under joint Navy/USAF sponsorship as the Texas Instruments' Multi-Purpose Radar/Missile Site Location System. It is scheduled for flight testing in late FY81-early FY82. Results of the far-term ASRAT phase will not be available in time for use in the CMMR program.

3.2.1.5 Advanced Fighter Technology Integration (AFTI)—63205/2506, 63245/2061

The AFTI program is still following the schedule noted in our previous report and there are no changes in our recommendations regarding this effort. The AFTI work, although not directly related to the CMMR program, requires coordination with the ASRAT efforts, which in turn could impact CMMR if additional funds for ASRAT are provided. There is little applicability of this program to CMMR at this time. However, the digital flight control/fire control developments may require a relook at a later time.

3.2.1.6 Low Altitude Navigation Targeting Infrared For Night (LANTIRN)—63249/2693

LANTIRN is scheduled to have production hardware available in December 1984 instead of December 1983 as indicated in our previous report. LANTIRN is still programmed for integration into the F-16 as an under-the-weather manual terrain-avoidance (TA) system. We see no impact on CMMR at this time. However, a slip in this program could establish a requirement for a CMMR TA demonstration during the flight test program.

3.2.1.7 Assault Breaker (PAVE MOVER)—63747/2217

The Assault Breaker (PAVE MOVER) program is following the schedule indicated in our previous report. The technology output of this program will not be directly applicable to a CMMR slow moving target indicator (MTI) capability because the PAVE MOVER concept involves a stand-off airborne platform, a large (3-meter) antenna, and ground processing.

3.2.1.8 Advanced Medium Range Air-to-Air Missile (AMRAAM)—63370/2437

The AMRAAM program is on schedule as indicated in our previous report. MIL-STDs-1553B, -1750A, and -1589A have been directed. MIL-STD-1760, currently in

*It is our understanding that this program might be cancelled.

draft form, could also be directed for AMRAAM if it is approved. As mentioned previously, the CMMR design must be responsive to the data requirements of AMRAAM and should not limit its operational capabilities with respect to range, track while scan, illumination envelope, etc.

3.2.1.9 Electronically Agile Radar (EAR)—63241/1206

Our recommendation in the previous report was to review EAR as a potential technology source for PSP hardware and software development occurring under the F-16 advanced radar program. While the EAR program office is closed, the Air Force Wright Aeronautical Laboratories (AFWAL) have accumulated data and algorithms for some modes that could be applicable to CMMR.

3.2.2 Status of Other Programs Based on Software Technology

3.2.2.1 Integrated Digital Avionics (IDA)—64219F

The IDA program is designed to establish and promote avionics architecture standards for use in new and retrofit weapon systems. To accomplish this objective, the program will provide the facilities and personnel required to develop, implement, control, and maintain avionics standards. The program, a new start in 1980, is the engineering-development continuation of its predecessor, the Digital Avionics Information System (DAIS) advanced development program.

IDA has incurred a funding cut for FY81, and its schedule will be slowed. However, there are facilities that can provide the capabilities necessary for implementing existing architecture and software standards. These include the ASD/ENASD Systems Engineering Avionics Facility (SEAFAC), the ASD Computer Center, and the Avionics Systems Analysis and Integration Laboratory (AVSAIL). The following are individual initiatives under this program.

Systems Engineering Avionics Facility (SEAFAC)

SEAFAC is designed to provide systems engineering support for avionics hardware and software. It serves as the MIL-STD-1750 Control Facility and is the custodian for MIL-STD-1553. SEAFAC support to the CMMR Program Manager is available in several areas. At the Program Manager's option, the -1553 multiplex group of SEAFAC could support CMMR in specification writing, proposal evaluation, documentation review, preliminary design review (PDR) and critical design review (CDR) support, and testing and evaluation. Additionally, if -1553 is directed for CMMR, SEAFAC would be required to validate CMMR hardware and software for compliance with that Standard. In FY82, SEAFAC is programmed to be the MIL-STD-1750 validation facility also. In this capacity, it will evaluate all -1750 computers to insure that the development contractor complies with the Standard.

A "hot bench" is also available at SEAFAC. This capability could be employed to perform CMMR feasibility studies and implementation testing should the Program Manager determine that the development risk, which could be affected by the CMMR direction, is such that this is desirable.

ASD Computer Center

The ASD Computer Center has been designated as the Air Force's new JOVIAL Language Control Facility. This facility would be required to validate any new J73 compilers developed for CMMR if J73 is directed. Other technical assistance areas available from the Computer Center include assistance in using support software tools, hosting and rehosting, source selection of support software contracting, and J73 compiler expertise.

Avionics Systems Analysis and Integration Laboratory (AVSAIL)

The AVSAIL facility is a full system simulator for evaluating developmental subsystems in a simulated environment. This facility employs a J73 compiler (DEC 10 host, MIL-STD-1750 target). Although AVSAIL is primarily a laboratory facility, if the CMMR software contractor has, or is supplied with, a DEC 10 machine and J73 is directed, the Program Manager could conceivably use AVSAIL's support software to assist in the development and test of CMMR operational software.

Interface Standards

In addition to the three Standards mentioned previously, IDA is currently developing a software interface Standard prescribing an executive-to-application functional interface. This system-integration-level Standard is undergoing industry review and could possibly be directed for CMMR.

3.2.2.2 Modular Automatic Test Equipment (MATE)—63247

The MATE program is designed to provide the Air Force with a family of common hardware and software test modules that can be used to build stations for unique test and support requirements. The ultimate goal of MATE is to establish the architecture and standard interface for test modules, which then can be built by industry in a competitive environment. This concept will also accommodate advancing technology while keeping the same interface standards. The program is in the advanced development phase with two competitive contractors, Sperry and Westinghouse. The winner of this phase will proceed into engineering development, which is scheduled to occur from mid FY81 through early FY85. It is quite possible that MATE could be directed for application to CMMR, with interim contractor support being supplied until MATE hardware is available. The LANTIRN program, with production scheduled for December 84, is currently under this direction. For LANTIRN, the initial test requirements will be defined at PDR with full definition coming several months later. The MATE program office will then decide which MATE requirements it can fulfill. Interim contractor support for LANTIRN is planned for the first 1 to 3 years at the Intermediate level and for the first 3 years at Depot level. Should CMMR receive similar direction, this aspect of the program would require careful analysis of interim support alternatives (e.g., reliability improvement warranty (RIW), RIW/mean-time-between-failures guarantee, or conventional interim contractor support) by the Program Manager.

3.3 REVIEW OF AIR FORCE AND NAVY SOFTWARE DEVELOPMENT APPROACHES

Both the Air Force and Navy have invested heavily in advanced and engineering avionics development over the years. One of the two candidate radars was developed primarily under Navy sponsorship, the other under Air Force sponsorship. This section makes

observations on these developments relevant to CMMR. Three major aircraft programs were examined: F/A-18, F-16, F-15. A portion of the information was gathered through field trips and interviews with knowledgeable managers and hardware and software engineers identified in Appendix B, while most was obtained from documentation listed in Appendix A. To provide consistency and to accelerate the interview process, a program checklist was developed (see Appendix C).

3.3.1 Program Management Approaches

The Navy's major avionics efforts, managed by the Naval Air Systems Command, are developed under a matrix organizational arrangement. A Program Manager obtains much of his personnel resources from hardware and software functional organizations that provide him full-time technical support. Thus, the Navy's program offices contain a relatively small number of individuals backed up by a large force of functional expertise. Furthermore, the Navy Program Manager is responsible for the system throughout its entire life. Such responsibility for continued support after the weapons system becomes operational gives him the opportunity to trade-off acquisition, support costs, and schedules during the early stages of a program. For example, in the F/A-18 development program, the Program Manager will have a very large cadre of support personnel at China Lake Naval Air Station to maintain the avionics software.

The Air Force Systems Command (AFSC) establishes a Systems Program Office (SPO) for its major acquisition endeavors. During full-scale engineering development, a SPO could have as many as 300 people working for a SPO Director (SPD). If the program is a major effort reviewed by the Defense Systems Acquisition Review Council, the Director probably would be a General Officer with his own personnel for engineering, support equipment, contracts, configuration management, etc. A recent reorganization at ASD formed the Deputy for Tactical Systems (ASD/TA). Formation of this new matrix-type organization indicates a trend toward the Navy approach for smaller programs—there is a smaller SPO cadre for the F-15, A-10, and Maverick programs, for example. Since the Air Force has a separate Logistics Command (AFLC), the AFSC acquisition work transitions to AFLC at an agreed upon Program Management Responsibility Transfer (PMRT) date, in accordance with AFR 800-19.

AFLC's Air Logistic Centers (ALCs) are responsible for the final hardware and software maintenance and operational support of a particular weapons system. A recent Air Force trend is to involve the ALCs in assisting in some of the program's software support aspects well before PMRT date. In the case of the F-16 program, for example, Ogden Air Logistics Center has been directly involved in the verification and validation of some of the F-16 avionics software programs and its Avionics Integrated Support Facility (AISF) is almost operational, four years ahead of the planned PMRT date. Current planning is to transfer software program responsibilities incrementally, as the Center achieves the required support capability.

Knowledge of these different management techniques and organizational relationships can help the CMMR Program Manager plan his program office structure and accommodate any interservice program needs.

3.3.2 Avionics Software Technology Investment Approach

The Air Force invests heavily in separate avionics software technology work, such as the IDA and MATE programs discussed earlier, while the Navy might pursue similar efforts

within a major program. The AFWAL/AA and RADC compiler effort to control the proliferation of the JOVIAL HOL (which is not tied to any specific weapons system program) is a good example of the Air Force's approach. The AFAL's previous DAIS work, which led to the development of MIL-STD-1750A and its associated prototype hardware (the AYK-15A), and the ASD/EN follow-on engineering development work (SEAFAC), are other examples. The common PSP work under PE 64201/2519 mentioned before is a fourth example. Although there are similar, and in some cases identical, technology program elements in the Navy and the Air Force, the Air Force is the lead service in many instances and, thus, controls more funds.

3.3.3 Computer Standardization

The Navy has installed the first standard avionics computer (the AYK-14) in the F/A-18. This computer is an outgrowth of the standard, general purpose, ground-based, mini-computer program (the UYK-20) initiated in 1973. As a result of exercising rigorous software and hardware configuration management controls, programs written for the UYK-20 can execute on the AYK-14, and programs written for the AYK-14 can execute on the Navy's next series of standard general purpose computers, the UYK-43 and UYK-44.

By taking this approach, the Navy has standardized on support software at least for one ISA—the MTASS 20/14 (Machine Transferable AN/UYK-20, AN/AYK-14 Support Software)—which enables aircraft to interface with shipboard and shore-based command and control systems. Paralleling its standard computer hardware development efforts, the Navy also uses, whenever possible, the CMS-2 HOL version "M" for its UYK-20 and AYK-14 computers; it uses version "Y" for other standard computers, such as the UYK-7.

The USAF emphasizes standardization for computer architecture and instruction sets rather than for computer hardware. The -1750 computer approach apparently has broader industry support than the Navy's, because it appears less constraining in terms of design innovation and manufacturing sources. Even though computer software developed for the AYK-15A might not operate as efficiently with this "common architecture machine" as software developed for a machine with a specific requirement in mind, its potential life-cycle-cost advantages appear extremely attractive.

3.3.4 Software Support

In order to more clearly understand the Navy and Air Force policies and procedures in this area, it is necessary to recognize differences in support requirements. The Navy must be self-sufficient while at sea for prolonged periods. Its development and operational phases of software support are separated by a transition phase that terminates on the Software Support Date (SSD).

The Naval Air Systems Command designates the necessary development and support activities very early in the life-cycle of the system. During the development phase, the Software Support Activity (SSA) participates to ensure that developed software can be organically supported. The SSA begins planning for support of a system at the initiation of Full Scale Development. In the transition phase, the SSA acquires facilities and trains personnel. The SSA assumes software support responsibility on the SSD. Coordination between the SSA and the operating forces is the responsibility of a Software Change Review Board (SCRB) in regard to all software changes or enhancements made subsequent to the SSD. SSD occurs at or after the Initial Operational Capability (IOC) date when the

contractor-developed software is sufficiently mature to allow transition of design and configuration control from the development contractor to the Navy. The primary mechanism by which life-cycle support is coordinated is the Software Life-Cycle Management Plan, similar to an Air Force CRISP.

For the Air Force, the time between aircraft weapon system life-cycle stages is more clearly defined due to the required transition from AFSC to AFLC. Three stages are used. Actual research and development occurs during the developmental stage. The transitional stage begins at IOC with software support provided by the developing agency (ASD) and moves to the operational stage at PMRT date. Software support planning begins with the formation of the CRWG. Chairmanship of the CRWG transfers from AFSC to AFLC at the PMRT date.

Before the system is operational, the support procedures for the avionics system are documented in an Operational/Support Configuration Management Procedures document, which describes in detail the working relationship between the supporting and using commands. For avionics, changes during development are an ASD responsibility. Close coordination between ASD and the designated ALC is maintained to ensure responsiveness to change requirements. ASD may be involved again if system changes require new research and development. In any case, it is an ASD task to equip the ALC with all software design data and to assist in acquisition of required assets to allow full support of the avionics system software by the PMRT date.

One of the CMMR Program Manager's initial software efforts will be to form the CRWG and develop the radar CRISP.

3.3.5 Digital Growth Philosophy

Both the Navy and the Air Force are involved in either new or major digital avionics upgrade programs. These programs reflect the results of years of intensive computer hardware and software technology efforts. Their new avionics suites parallel the explosion that has occurred in the commercial marketplace.

The F-15 avionics was selected for review because it is just now "evolving" with the introduction of the new digital radar PSP mentioned earlier. This PSP, built by Hughes, is similar to that of the F/A-18s. The F-16 was selected for review because it will have one of the CMMR candidate radars and is planned to be the testbed aircraft for the CMMR program as discussed in our previous report. The F-16 is also being upgraded with several modifications, either underway or planned. These modifications are discussed in subsection 3.5.5 and Appendix D. The F/A-18 was selected for review because it is the latest development in this class of aircraft and contains the other CMMR candidate radar, the APG-65.

The purpose of studying the status of the avionics architectures of these three aircraft is to provide insight into the impact of installing common radar hardware/software on the CMMR candidate aircraft and to assess the level of compliance with the software guidelines reviewed in Task 1.

3.4 F-15 SOFTWARE DEVELOPMENT APPROACH AND COMPLIANCE WITH SOFTWARE GUIDELINES

An overview of the F-15 avionics architecture is contained in Appendix D. Its software consists of three operational flight programs (OFPs), all written in assembly language. One is for the Central Computer (CC), one for the Radar Data Processor (RDP), and one for the Radar Warning Receiver (RWR). The details of logistics planning for all F-15 software is contained in five CRISPs (avionics, automatic test equipment, Tactical Electronic Warfare System (TEWS), Operational Flight Trainer, and Maintenance Trainer). The CC and RDP OFPs are supported by MCAIR. Hughes supports the RDP OFP under subcontract to MCAIR. The RWR OFP is maintained by the Warner-Robins Air Logistic Center EW support facility. The overall software development approach for the F-15 is Prime Integrating Contractor (PIC).

3.4.1 Central Computer OFP

The CC OFP is modularized, which means that each of its program modules can be independently developed, debugged, and tested. Changes can be made within a module or by adding additional functions without significant change to the other modules, provided the external interface remains the same. The eight CC OFP modules are:

- Executive
- Air-to-air
- Air-to-ground
- Navigation
- Flight director
- Controls and Displays
- Self Test
- Math Subroutine

3.4.2 Radar Data Processor (RDP) OFP

The RDP (Computer) OFP is also modularized and has eight program modules:

- Executive
- Power Up
- Antenna Control
- Math Subroutine
- Search and Acquisition
- Track
- Displays
- Built-in Test (BIT)

The radar hardware and software are being update by Hughes under subcontract to MCAIR. Existing OFP modules will undoubtedly change. The CMMR Program Manager is encouraged to track the radar OFP effort closely, especially in algorithm development of the new Air-to-Ground modules for application to CMMR.

3.4.3 Radar Warning Receiver (RWR) Software Elements

The software "elements" in the F-15 TEWS are the OFP in the RWR computer, a set of programmable read only memories (PROMs), which are used in conjunction with the Internal Countermeasures Set (ICS), and a set of PROMs used for character generation in the RWR display.

3.4.4 Current Radar and Upgrade Programs

The Fire Control Radar Set installed in the F-15 is the Hughes AN/APG-63. It is a coherent, X-band, multiple pulse repetition frequency, multi-mode attack radar, which has the ability to search for, acquire, and track airborne targets and provide air-to-ground mapping and ranging. The APG-63 provides rapid acquisition of short range airborne targets and head-on capability against long-range, high-closing-rate targets. It also provides information to other avionics and to the pilot to increase weapon effectiveness. The radar was procured as contractor-furnished equipment from MCAIR.

The F-15 radar is being modified under ECP 937 to change its hardwired signal processor to a programmable one for advanced capability and enhanced employment with AMRAAM. Since most of the APG-63 functions and modes remain essentially the same, the radar will not be redesignated. The PSP upgrade began in production aircraft in May 1980 and will affect 325 aircraft. Some C and D aircraft will require retrofitting. As previously reported, this change also adds new raid assessment, doppler beam sharpening, and ECCM modes.

Replacement for the original 16k × 24-bit Hughes core memory radar computer (HCM-231) for all A and B aircraft with a Hughes solid state 24k electrically alterable read only memory (EAROM) began in 1978. PSP aircraft will have new 96k expandable solid-state computers for non-volatile program storage for the PSP as well as for RDP OFP expansion. In both the 24k and 96k machines, the original 4k solid state random-access memory for scratch-pad use is also being upgraded to 16k. The 24k computer memory is full, while the 96k computer will have approximately 30k of spare memory.

The new, special purpose PSP is similar to that of the Navy's F-14 and F/A-18 aircraft. Fifteen of the 21 hardware modules or shop replaceable units (SRUs) are identical. This PSP can perform up to 7.2M operations per second—much faster than general-purpose computers. Both the RDP and PSP have been designed for expansion if needed.

While there are hardware similarities in the F-15 radar PSP and in one of the CMMR candidates—the F/A-18 APG-65—the extent of the software similarities was difficult to assess because of the status of the documentation on the later aircraft. The extent of hardware and software support savings that could be realized by utilizing existing Air Force and Navy support resources for the Hughes' radar system is, therefore, undetermined at this time.

3.4.5 F-15 Compliance with Software Guidelines

The F-15 was the first Air Force aircraft to employ a digital data bus, and was designed and built before MIL-STD-1553 was prepared and directed for Air Force use. Similarly, MIL-STD-1589 and -1750A were developed after the aircraft was in production. As a result, the aircraft does not conform to these relatively new Standards.

3.5 F-16 SOFTWARE DEVELOPMENT APPROACH AND COMPLIANCE WITH SOFTWARE GUIDELINES

The F-16 avionics architecture is presented in Appendix D. The overall software development approach for the radar through tail number 250 is to use the aircraft manufacturer, General Dynamics, as the PIC.

3.5.1 Software Approach

The F-16's aircraft's software consists of the eight OFPs listed in Table 3-2 along with their host computer.

The logistics planning for all F-16 software is contained in a system-level CRISP (with a series of seven appendices). These documents may be reviewed for more detailed computer resources logistics support information and more knowledge of the software support concepts, organizational responsibilities, and detailed procedures. The seven avionics OFPs are the responsibility of General Dynamics, while the RWR OFP is maintained by the Warner Robins Air Logistics Center EW support facility.

Initially, in-house support is planned for the Fire Control Computer (FCC) and Stores Management Set OFPs. A dynamic system simulator and an avionics equipment bay have been procured to support the FCC OFP, as well as equipment necessary for the Stores Management Operational Program.

The use of in-house software support for the Fire Control Radar and Inertial Navigation System is still under investigation. Since these systems are under a Reliability Improvement Warranty program, an in-house capability will not be needed until CY83.

The Head-Up Display, the Radar Electrical-Optical Display, and the Central Air Data Computer OFPs are to be supported by the respective vendors. Projected program change rates do not currently justify an in-house capability for these firmware systems. However, the F-16 SPO is acquiring the necessary documentation to establish an organic capability in case such an approach becomes more cost-effective or necessary to meet operational support requirements.

For planning purposes, organic OFP update "block" changes are scheduled approximately once per year. However, user priorities and projected AFLC workload can affect the detailed scheduling, as well as the anticipated time span to retrofit released OFP updates. To date, there have been two major block changes to the F-16 avionics software.

3.5.2 Fire Control Computer OFP

The FCC OFP provides the logic and computations necessary to implement and integrate fire control system modes and functions. The OFP consists of computer processing

Table 3-2. F-16 OPERATIONAL FLIGHT PROGRAMS

Fire Control Computer	(Delco Magic 362F-2)
Inertial	(SKC-3000)
HUD	(Marconi)
Fire Control Radar Computer	(Westinghouse)
Stores Management Set	(GD-8080)
Radar/EO Display	(Kaiser)
Central Air Data Computer	(Sperry)
Radar Warning	(CM-479)

instructions that have been developed to satisfy allocated avionics requirements. Because of its central role in integrating F-16 sensors and equipment into the desired fire control system, the OFP is designated a configuration item and meets the configuration management requirements of MIL-STD-483 and the Configuration Management Plan (16PP153).

The FCC OFP is a real-time program, that coordinates sensor and equipment data transfers over the -1553A data bus. It schedules various processing activities to implement the fire control and navigation modes selected by the pilot.

The FCC OFP was developed by General Dynamics, while its executive module was written by Delco. Most of the processing instructions (80 percent) that comprise the OFP are written in the J3B-2 HOL to support advanced concepts of software documentation, understanding, and maintainability. Use of the J3B-2 HOL also has facilitated modular design and testing. In the design process, each functional requirement has been mapped into one or more OFP components for implementation. The definition of components is accomplished through the top-down, structured programming methodology, which results in a linear, modular program with readily identifiable hierarchical levels and single entry and exit points for each module. As a result, the OFP can be easily read and tested, and revisions to the OFP can be easily undertaken and accomplished.

Contractual specifications provide for 30 percent memory and 40 percent speed reserves in the FCC OFP system although the memory reserve is presently only about 20 percent. Detailed data on the requirements for this program and their implementation may be found in the software development and product specification documents (16ZEO11-1 and 16AEO11-2).

Table 3-3 lists the 15 major FCC modules. Computer storage allocations indicating an 80 percent memory fill in accordance with the Part II Product Specification, dated 15 May, 1980, are also shown in the table.

Table 3-3. F-16 FCC OFP MODULES AND COMPUTER STORAGE ALLOCATIONS

Module	Storage Allocation (Number of Words)
Executive	273
System Control	3324
Bus Control	2305
Initialization and Error Handling	553
Air-to-Air Gunnery	514
Air-to-Air Missles	2243
Air-to-Ground Attack	2728
Stores Data Select	1859
Data Entry and Display	2321
Cruise Energy Management	618
Combat Energy Management	940
Navigation Support	769
Fix Taking	2703
Self Test	1582
Support Utilities	787
Common Pool Storage (Data Areas)	1633
Stack Area	863
	26,015

3.5.3 Fire Control Radar OFP

The radar computer OFP is written in assembly language and executed on the fixed-point Westinghouse "MILLI" Fire Control Radar computer. The OFP design is structured module (29 functional tasks) but not structured programming. It provides the processing necessary to implement F-16 radar system modes and functions and provides overall control of the radar hardware. The radar operating mode is determined by mode commands received from the Radar Control Panel, the Fire Control Computer, and the Stores Management System. The Fire Control Radar OFP is undergoing a second major update to its 30 January, 1979, baseline.

An advanced radar OFP will be written in the J73/I HOL initially, then in J73. A J73 computer hosted on Westinghouse's VAX 11/780 and targeted for their new MILLI-EP computer will be available in about a year. The advanced radar is being designed with a new floating point computer, an array processor, and 256k block oriented random access memory (BORAM), all contained in one line replaceable unit (LRU) to fit in place of the existing computer and digital signal processor. The use of BORAM in the architecture of the advanced radar resembles the Navy/Hughes APG-65 approach.

3.5.4 Current Radar

The multimode F-16 radar, built by Westinghouse under subcontract to General Dynamics, is a pulse-doppler system operating in the x-band. The unit weighs 270 pounds and has a volume of four cubic feet. It consists of an antenna, a transmitter, a low-power radio-frequency unit, a digital signal processor, a control computer, and a radar control panel. All six items are LRUs.

The radar provides both air-to-air and air-to-ground modes of operation. In the air-to-air mode, the radar can be configured to provide either normal PRF look-up capability or a medium PRF look-down capability for airborne targets located below the aircraft. In the air-to-ground mode, the system is configured to be a low PRF radar. Air-to-ground ranging, ground map, ground map doppler beam sharpening, and beacon map modes are provided.

It should be noted that beginning with aircraft number 250, Westinghouse will deliver the APG-66 radar as government-furnished equipment (GFE) to General Dynamics. For the purposes of the analysis in Chapter 4, this acquisition approach has been termed "Radar Integrating Contractor" (RIC). A similar acquisition approach will be followed for the advanced radar.

3.5.5 Advanced Radar

The F-16 advanced radar is at the prototyping stage under a Westinghouse contract to provide a new PSP and a more powerful transmitter to increase the range to a level compatible with AMRAAM. This new radar will be used as one of the CMMR candidates in the dual-source fly-off.

Phase 1 of the advanced radar program will demonstrate new capabilities (discussed in Appendix D) and build in growth in preparation for Phase 2 and production. Phase 2 will add several new modes to the OFP but primarily will qualify the software and hardware.

A common radar specification and F-16/APG-65 initial integration studies have already been accomplished by the F-16 SPO to prepare the way for the dual source fly-off. With approval of the advanced radar program, first-unit production delivery could occur in October 1983.

3.5.6 F-16 Compliance with Software Guidelines

The F-16 employs a MIL-STD-1553 digital data bus. A -1553B display bus is planned. The FCC employs the J3B-2 HOL for approximately 80 percent of its OFP. The present radar OFP is written in assembly language. An advanced radar OFP is under development with J73/I as its source code. During Phase 2 of the program, the OFP will be upgraded to MIL-STD-1589A. There are no plans to employ MIL-STD-1750A hardware in the radar.

3.6 F/A-18 SOFTWARE DEVELOPMENT APPROACH AND COMPLIANCE WITH SOFTWARE GUIDELINES

An overview of the F/A-18 avionics architecture is contained in Appendix D. The F/A-18 software consists of five OFPs written in assembly language, except for the mission computer OFP which has a mix of CMS-2M HOL and assembly language. The other four OFPs are for the radar, inertial, stores management, and air data computer subsystems. Although not classified as OFPs, additional programs are contained in other F/A-18 airborne processors such as the Flight Control Electronics, Communications, Armament Monitor and Control, and Multi-Function Display Group subsystems.

The logistics planning for all weapon system, ground support equipment, and trainer software is contained in the F/A-18 Software Life Cycle Management Plan, a document similar to the Air Force's CRISP. Total weapon system software responsibility is the responsibility of MCAIR—the PIC approach.

3.6.1 Mission Computer OFP

The Mission Computer OFP resides in two AYK-14s; it is broken down into 23 modules as indicated in Table 3-4. Even though these modules are similar to those of the F-15 and F-16 aircraft discussed earlier, the OFP is "multi-role" and does not change for either the fighter or attack mission. The back-up modules allow continuation of the mission should one of the main computers fail. About 20 percent of the OFP is written in CMS-2M HOL; the balance is in assembly language.

3.6.2 Radar OFP

The radar OFP is written in assembly language and stored in a disc memory. Readout is to a 16k solid state working storage. Its design is structured module and is broken down into the instructions for the RDP and the RSP. The projected storage allocation for the radar disc is shown in Table 3-5. It should be noted that the instructions are 48 bits for the RSP and 16 bits for the RDP.

3.6.3 Current Radar

The F/A-18 radar, built by Hughes under subcontract to MCAIR, is an X-band, mono-pulse, coherent, multimode, multimission system designed for both air-to-air and air-to-surface fighter and attack roles. The unit weighs 340 pounds and has a volume of 4.5

Table 3-4. F/A-18 MISSION COMPUTER OFP MODULES

AYK-14 Weapon Delivery Computer (9 Modules)	AYK-14 Navigation Computer (14 Modules)
Executive	Executive
Air-to-Air	Navigation
Air-to-Ground	Navigation HUD
Weapon Delivery HUD	Engine Monitor
Tactical Controls/Displays	Navigation Controls/Displays
Self Test	Support Controls/Displays
Math Sub Routines	In-Flight Monitor and Recording
Data Base	Avionics BIT
Navigation Back-up	Non-Avionics BIT
	Self Test
	Math Sub Routines
	Data Link
	Data Base
	Weapon Delivery Back-up

Table 3-5. PROJECTED MEMORY REQUIREMENTS FOR APG-65 (WORDS)

Radar Data Processor	
Air-to-Air	33,935
Air-to-Ground	18,455
Built-in-Test	17,225
Total	69,615

Radar Signal Processor	Instructions (48 bits)	Constants (24 bits)	Disc Storage (16 bits)
Air-to-Air	12,108	1,576	38,688
Air-to-Ground	13,749	2,212	44,565
Built-in Test	4,051	0	12,153
FFT Coefficients	0	1,024	1,536
Executive and Loader	520		1,560
Total	30,428	4,812	98,502

Projected Disc Usage (250k Capacity)	
RDP	69,615
RSP	98,502
Flight Test Memory	4,000
Patch Spares	5,883
Total	178,000

cubic feet. The radar consists of an antenna, transmitter, receiver-exciter, data processor (computer), and signal processor. All five items are LRUs. Other features of the APG-65 were discussed in our previous report.

The RDP interfaces with the Mission Computers and commands the operating modes, provides timing and mode constants, handles radar fault detection, and manages avionics and display processing. It also provides the dynamic parameters, threshold, and AGC constants. The RSP provides target hit and miss, range, velocity, tracking, and discriminant data to the RDP, and analog video data to the display. The RSP also performs timing for such functions as:

- A/D sampling
- Pulse compression
- Transmit-receive keying
- Receiver gain control.

Since its original design, the APG-65 software has grown considerably. Recent changes in the hardware brought about by this growth have not had a significant impact on the program, since the baseline has not been frozen and the first radar production system was only delivered recently (June 1980), after 14 full-scale development systems.

The software and hardware characteristics of the APG-65's RDP, RSP, and disc memory are summarized in Table 3-6.

Table 3-6. SOFTWARE AND HARDWARE CHARACTERISTICS OF APG-65 RADAR

	Radar Data Processor (RDP)		Radar Signal Processor (RSP)	
	FSD/PROD	PROD (3rd LOT)	FSD/PROD	PROD (3rd LOT)
Memory (Disc)	150k	250k		
Memory (Core)	16k	32k		
Memory (RAM)			48k	196k
I/O Buffer (RAM)	4k	16k		
Scratch Pad (RAM)	4k	16k		
Memory Fill	100%	60%	100%	90%
Word Size	16 Bit		16 Bit	24/48 Bit
			Equiv. Usage	
Speed	700k ops		7.2M ips	
Manufacturer*	Hughes		Hughes	

* Disc—Sperry Flight Systems

3.6.4 F/A-18 Compliance with Software Guidelines

The F/A-18 employs a MIL-STD-1553A digital bus. The two Mission Computers contain the overall weapon system OFP, 20 percent of which is written in the CMS-2M HOL. There are no plans to use a HOL for the radar OFP or employ MIL STD-1750A ISA for any of the avionics computers.

3.7 SUMMARY

3.7.1 Software Technology Programs

Our update review of nine radar technology programs to determine their applicability to CMMR software development revealed that only four have any significant impact on

CMMR. These are listed in Table 3-7 with relevant information:

Table 3-7. RADAR TECHNOLOGY PROGRAMS AFFECTING CMMR

Title	PE/Project	Status	Impact on CMMR
RPSP	64201/2519	Continuing	S/W algorithms transferable to F-16 if done ASAP
NCTR	63742/1177	Continuing	Algorithms transferable to CMMR
ERIP	64201/2259	Cancelled	Terrain following radar portions applicable to CMMR
EAR	63241/1206	Project completed	Technology source for some algorithms and software

3.7.2 Production Aircraft Programs

Table 3-8 summarizes the important features, status, and differences of the avionics architecture, software, and radar hardware of the three weapons systems reviewed.

The following are our more important findings that are used as inputs to the Chapter Four analysis:

- The radar hardware and software designs for all three aircraft investigated are still being updated; the radar software probably will not mature for several years. It is possible that the F/A-18 radar will mature sooner than that of the F-15 or F-16, since both the F-15 and F-16 radar updates are more recent. A further increase

Table 3-8. Characteristics of the Three Weapons Systems Reviewed

Weapon System	Avionics Architecture	Avionics Software	Radar Hardware
F-15	Stable • Central computer (16k) • First digital mux bus • Some hardwired systems	Changing • Two OFPs • Assembly language • EAROM use • Organic support planned	Changing • From hardwired DP to PSP • Larger computer • Two computer configurations
F-16	Changing • Fire control computer (32k) • First use of 1553A bus • New 1553B bus to be added • ECS in each major subsystem	Changing • Seven OFPs • FCC J3B-2/assembly language mix • EPROM/RAM use • Some phased organic support planned	Changing • Larger computer being installed • New PSP, Tx, computer, and array processor under development • New memory storage device (BORAM)
F/A-18	Stable • Two mission computers (64k ea.) • 1553A mux bus • ECS in each major subsystem	Evolving • Five OFPs • MC CMS-2M/assembly language mix • Core/Disc/RAM/USE • Organic support when possible	Changing • Larger disc memory • Larger computer • Larger signal processor

in the F-15's radar computer memory and the F-16's FCC probably will be required if new capabilities are added.

- Avionics hardware and software for all three aircraft have been designed for modularity. None of the aircraft use software structured programming for their avionics subsystems. The only standard hardware is the F/A-18 standard airborne computer (AYK-14). The modular design of the software has helped, but changes to other modules and OFPs have been required.
- Although the memory requirements for the central, fire control, and mission computers of the three aircraft have remained stable, the radar memory requirements for all three aircraft have increased by factors ranging from 3 to 6. These increases are the result of software design changes due to deficiencies in design and new software requirements. They have resulted in considerable change to the radar OFPs and made minor impacts on other OFPs. Fortunately, in each case, the radar hardware changes have been accommodated in the space available without much difficulty.
- The Air Force has been the first to use solid state memories (EPROM, EAROM, RAM, ROM for the F-15 and F-16 avionics). The Navy primarily uses core memories for its subsystems; it uses a disc for storage of the radar OFP.
- The impact of solid state memories on supportability and maintainability of an OFP was not reviewed. The Air Force has no field load capability for an F-16 OFP and plans to support these memories organically in about two years.
- The F-16 advanced radar's design concept is similar to that of the F/A-18s with a 256k BORAM for storage of the radar OFP. The Navy uses a 250k disc memory for storage of the APG-65 OFP.
- Both the Navy and the Air Force use HOLs for their main mission and fire control computer OFPs. The F-16's advanced radar OFP is being coded in the J73/I HOL (about 85 percent J73/I, 15 percent assembly language) with a planned update to J73. The APG-65 OFP is written in assembly language and there are no plans for updating to a HOL.
- Both the Navy and the Air Force are using a "phased support concept" for all OFPs. This involves initial contractor support with later transitions to organic support. The radar OFPs for all three aircraft are contractor supported by the aircraft manufacturer. In some cases, software support of certain OFPs might never go organic. In the case of the F-15/F-16 radar OFPs, the Air Force decision on support is still pending. They might remain contractor maintained with limited Air Force organic changes for electronic counter countermeasures. The Air Force is delaying a decision on the support concept until more knowledge becomes available concerning these OFPs and their target computers, the degree of difficulty anticipated in maintaining them, and availability of the skills needed to support them.

A summary of the overall development approach, current status and current compliance with the key Standards reviewed in this chapter (-1553A/B, -1750A, -1589(A), ATLAS) is presented in Table 3-9.

Table 3-9. ASSESSMENT OF CURRENT PRODUCTION PROGRAMS

Weapon System	Development Approach	Current Status	Current Compliance with Key Software Standards
F-15	• Prime Integrating Contractor • Centralized Architecture • First digital bus • Extensive use of solid state memories • Assembly language source code	• New radar computer & PSP	None
F-16	• Prime Integrating Contractor • Decentralized architecture • First 1553 bus • Extensive use of solid state memories • Partial use of a HOL	• New radar under development	Uses: -1553 -1589 HOL(FCC) -ATLAS
F/A-18	• Prime Integrating Contractor • Distributed architecture • Extensive use of core memories • Partial use of a HOL	• Baseline not yet frozen	Uses: -1553A -CMS-2M HOL (Mission Computers)

CHAPTER FOUR

ALTERNATIVE CMMR SOFTWARE ACQUISITION STRATEGIES (TASK 3)

4.1 INTRODUCTION

Using the results of Tasks 1 and 2 and our previously developed hardware acquisition strategies covered in ARINC Research Report 1564-11-2-2122 (see page 1-1), we have evaluated the impact of alternative CMMR software acquisition strategies on operational capability, cost, schedule, risk, supportability, and management. This chapter reviews the potential advantages and disadvantages of alternative software acquisition approaches and highlights some related support considerations.

4.2 ALTERNATIVES

Three alternative software acquisition strategies were developed and analyzed: responsibility assigned to (1) a prime integrating contractor (PIC), (2) a radar integrating contractor (RIC), or (3) a single operational flight program developer and integrating contractor (SIC). Any one of the three strategies would begin approximately six months before the winner of the dual fly-off between Hughes and Westinghouse is declared. It is assumed that the market size will have been established and user requirements firmed-up by this time and that the market will sustain two or more manufacturers.

4.2.1 Prime Integrating Contractor (PIC)

The PIC approach assumes that there would be five individual contractors responsible for the radar OFPs required for the five candidate aircraft (see Table 2-1). These contractors are assumed to have a better overall knowledge of individual aircraft avionics but would require assistance from the radar manufacturer to integrate the radar hardware and its OFP into the weapon system. The radar could be supplied either as GFE or CFE.

4.2.2 Radar Integrating Contractor (RIC)

The RIC approach assigns responsibility to the radar hardware manufacturer for the hardware production, radar OFP development, and integration of the radar and its OFP into the avionics suites of each aircraft. Assistance from the developer of the avionics main computer hardware and software and the subsystems manufacturers would be required. Because of the quantities of units potentially involved, this approach would involve two or more radar manufacturers.

4.2.3 Single OFP Developer and Integrating Contractor (SIC)

In the SIC approach, one software contractor develops and integrates all radar OFPs. He would very likely require contractual assistance from both the avionics software and the radar hardware manufacturers. The SIC could team with the radar or avionics developer.

4.3 DEFINITION OF CRITERIA

Each of the CMMR alternative software acquisition strategies selected display certain advantages and disadvantages. Attention to the following five criteria was specifically tasked to insure that the Air Force and the CMMR Program Manager was given the best assistance in understanding the relationships among the alternatives chosen:

- Operational Capability
- Cost
- Schedule
- Risk
- Supportability

We have added a sixth criterion—Management.

4.3.1 Operational Capability

Operational capability is keyed to the ability of the contractor to meet the software OFP requirements of the radar hardware selected. It does not address the larger issue of overall weapons system operational requirements. The ability of the PIC, RIC, or SIC to meet the radar's OFP acquisition needs, including the weapon system avionics interface requirements, is paramount in establishing how well he meets this criterion. The prime factor in evaluating this criterion is the ability of the contractor to build growth and flexibility into the OFP design and accommodate threat changes in a timely manner.

4.3.2 Cost

Cost is a measure of the economic impact of the radar software acquisition approach selected. Our quantitative evaluation does not include, at the present time, costs for independent verification and validation and integrated avionics aircraft flight test time. We made an assumption that these costs would apply equally to all three software acquisition strategies in a comparison analysis.

4.3.3 Schedule

Schedule is a measure of the relative likelihood that the PIC, RIC, or SIC can meet the schedule demands of the five CMMR candidate aircraft. The integrating contractor selected will need to have an overall understanding and appreciation of the difficulties associated with developing a large radar OFP that can match the radar production hardware schedules with the overall weapons system test and integration schedules leading to the weapon system IOC.

The latest ASD CMMR schedule calls for the winner of the fly-off (Hughes or Westinghouse) to be selected in August 1982. First production delivery for the F-16 would occur two years later, with first production radar deliveries for the other candidate aircraft occurring in August 1985. For some of the candidate aircraft, such as the B-52, meeting the aircraft avionics' schedule will be extremely important.

4.3.4 Risk

In establishing the overall risk associated with the CMMR program, schedule and cost are excluded because they are addressed in other criteria. Risk evaluation is a subjective judgment of the relative difficulty of integrating the radar OFP into the rest of the avionics software under the PIC, RIC, and SIC concepts. For example, one radar hardware manufacturer alone cannot meet the production quantity demand for the radars required for all five candidate aircraft; a teaming or leader-follower approach would be required to meet schedules. When more than one hardware manufacturer is required, development and control of a "baseline" CMMR OFP becomes more difficult.

One of the key factors in the selection of the software acquisition strategy will be the software status of the F-16 advanced radar and the F-18 APG-65 OFPs at the time of strategy selection. Another factor will be the ease with which the OFP can be updated to include additional modes for the F-16 and other candidate aircraft in a timely manner.

4.3.5 Supportability

Supportability is a measure of how well the contractor can meet the demands of supporting the radar OFPs of the candidate aircraft initially and at the same time set the stage for organic support.

The avionics architectures for the DoD's newer aircraft are highly integrated; there are trends toward greater use of GFE; the responsibilities for OFP support and configuration control are shared among many different organizations; more than one OFP will be involved in the CMMR program. Clearly, there is no precedent for a military standardization program that impacts as many interfaces as does CMMR.

In our analysis of this criteria, we assumed that the software support will transition to organic as soon as possible.

4.3.6 Management

The Management criterion concerns the relative ranking of the management complexities involved in acquiring the CMMR software. For each of the three alternatives, there must be an assessment of how difficult it might be to define the responsibilities of various contractors and DoD organizations involved, the ability to control costs, and the degree of involvement to which the CMMR Program Manager must commit government resources.

4.4 ANALYSIS OF SOFTWARE COSTS

4.4.1 Approach

There is a dearth of actual acquisition data that distinguishes hardware from software costs in production aircraft. Cost data for development of OFPs that use HOL source code are even more rare. The only HOL applications are in the F-16 Fire Control Computer, which is under RIW, and the F/A-18 Mission Computers, for which the OFP baseline has not been frozen. We reviewed several ECPs for the F-16, and these provided valuable insight into the radar software mechanization. This review is presented in Appendix D, which describes the F-16 avionics architecture. We were not able, however, to obtain any useful cost data for the CMMR comparative analysis.

Since we could not locate comparable manufacturer's estimates or actual values for use in our examination of the cost impact of the acquisition alternatives, we turned to the cost estimating tools employed within the DoD to assist planners in developing software acquisition costs. An overview of the major tools is contained in Appendix E. One of the cost models, RCA's PRICE S, is used frequently in software acquisition cost estimating for major DoD programs. PRICE S is a DELPHI/parametric model proprietary to RCA. However, a derivative version employing the basic principals of PRICE S has been implemented by ASD/ACCX.* We selected this model because of its ease of implementation and use for parametric analysis. To date, the primary use of this model has been to assist ASD in source selection.

4.4.2 Assumptions

Use of the ASD model requires an estimate of the projected memory allocations for the CMMR OFP instruction and computer-associated radar software. We based these on the present F-16 Advanced Radar** and the F/A-18 APG-65† designs.

An assumption was made that the "baseline" CMMR OFP would have a 180,000, 16-bit instruction/word modular program during the fly-off; this would increase to 205k after the fly-off to accommodate the needs of the candidate aircraft. This compares closely with the 191k projected for the F-16 (including growth) and 178k projected for the APG-65. It was also assumed that the program is written in assembly language, since the model uses machine instructions (WORDS) and makes the assumption that the number of assembly and machine instructions are the same. The cost for the conversion of assembly language to J73 is analyzed in a later section.

The baseline CMMR OFP instruction distribution is indicated in Table 4-1. Using this baseline and analyzing the existing and desired radar modes of each of the candidate aircraft, we were able to develop instruction estimates for seven OFPs for the five candidate aircraft. These are listed in Table 4-2. The estimates were developed by the use of approximately 75 percent of the baseline modules and some new design and code, as indicated, for the cost model inputs. Note the assumption that the B-52 has two radar OFPs and the F-111D has a separate OFP because of its unique air/ground requirements. The specific radar modes were covered in our previous report, and are summarized in Tables 4-3 and 4-4. From these Tables, judgements have been made regarding the final OFP sizes. For example, in the case of the F-111D, its larger program size (relative to that of other F-111s) is attributed to its additional air-to-air and air-to-ground modes. In the case of the F-106, its small program size reflects its lack of air-to-ground modes.

4.4.3 Other Cost Model Inputs

In addition to basic OFP instruction input (I), the model requires a constant (C), exponent (e), and G&A, profit, and labor-rate inputs as discussed in Appendix F.

---

*_TI-59 Handheld Calculator Software Cost Estimating Model_, published by the Comptroller's Office, Aeronautical Systems Division (ASD/ACCX), Wright-Patterson AFB, Ohio June 1980. The major features of this model are described in Appendix F.

**Source: Advanced Radar PDR held at Westinghouse in June 1980.

†Source: APG-65 Program Review held at Hughes in June 1980.

Table 4-1. BASELINE CMMR OFP SIZE
(Modular program, written in assembly language)

Radar Computer	
Air-to-air Modes	35k
Air-to-ground Modes	23k
Built-in-test	17k
Subtotal	75k
Signal Processor	
Air-to-air Modes	43k
Air-to-ground Modes	62k
Built-in-test	12k
Miscellaneous	3k
Subtotal	120k
Miscellaneous	10k
Total	205k

Table 4-2. OFP STRUCTURES OF CANDIDATE AIRCRAFT

	F-16	FB/F-111 A/E/F	F-111D	B-52 Air/Air	B-52 Air/Grnd	F-4E	F-106
Final PGM size (k)	205	140	160	80	100	150	40
Initial PGM size (k)	180	105	125	60	75	115	30
New Code (k)	50	35	35	20	25	35	10
New Design (k)	25	5	5	5	10	5	<1

For the comparison of acquisition approaches, the G&A, profit, and labor-rate inputs were assumed to be the same for all OFPs (10 percent, 10 percent, and $5500/man-month, respectively). The G&A and profit parameters are those normally used by ASD; the labor rate has been escalated for inflation. Table 4-5, extracted from Volume II of the model description, provides the Complexity, Integration, and Resource parameters that establish the constant and exponent inputs for the model. These parameters are defined in Appendix F.

In the ASD model, the analyst also has the flexibility of adding costs for OFP integration into the avionics software. This additional integration factor was used to indicate the differences in the strategies selected.

The cost model parameters for each of the three acquisition strategies and the seven OFPs are listed in Table 4-6. By applying engineering judgment to selection of the parameters for the seven OFPs, we developed costs for the three strategies. For example, all software contractors "learned" through the use of the Resource parameter. We attributed an experience growth to the SIC after the first two OFPs and to all three software developers for the F-111D OFP. We also attributed an experience "credit" to the PIC in

Table 4-3. EXISTING AND DESIRED AIR-TO-AIR MODES

Air-to-Air Modes	Aircraft Types										
	F-18	F-16	F-15	F-111A	F-111D	F-111E	F-111F	FB-111	F-4E	F-106	B-52
Air Combat	X	X	X						D	D	D
• Rapid Search	X	X	X						D	D	D
• Auto Acquisition	X	X	X						D	D	D
•• Boresight	X		X		X						
•• Super Search	X	X	X								
•• Vertical Scan	X	X	X								
•• Auto Gun	X	X	X								
• Auto Track	X	X	X						D	D	D
Normal Air	X	X	X	X	X	X	X	X	X	X	X
• Look Up	X	X	X	X	X	X	X	X	X	X	X
• Look Down	X	X	X	D	X	D	D	D	D	D	
• Manual Acquisition	X	X	X	X	X	X	X	X	X	X	X
• Single-Target Track	X	X	X	X	X	X	X	X	X	X	
• Track While Scan	X	GP	GP						D	D	X
• Long-Range Search	X	GP	X								
• Range While Search	X	D	X								
• Velocity Search	X	D	X								
• Short-Range Search	X	D	X								
• Identification (IFF)	X	X	X		X				X	D	
• Noncooperative Target Recognition	GP	D	D						D	D	
• Raid Assessment	X	D	GP						D	D	
Target Illumination	X	GP	X		X				X	X	
Helicopter Detection		GP	GP						D		

X = Existing.
GP = Growth potential (possible with present hardware).
D = Desired (requires new hardware).

Table 4-4. EXISTING AND DESIRED AIR-TO-GROUND RADAR MODES

Air-to-Ground Modes	Aircraft Types										
	F-18	F-16	F-15	F-111A	F-111D	F-111E	F-111F	FB-111	F-4E	F-106	B-52
Ground Mapping	X	X	X	X	X	X	X	X	X		X
• Spoiled Beam	X				X						
• Real Beam	X	X		X	X	X	X	X	X		X
• High Resolution	X	X	GP	D	X	D	D	D	D		D
•• Doppler Beam Sharpening	X	X	GP		X						
•• Synthetic Aperture Radar	X										
• Freeze	X	X		D	X	D	D	D	D		D
• Expand	X	X			X						
Air-to-Ground Ranging	X	X	X	X	X	X	X	X	X		
Ground Moving Target	X	GP	GP	D	X	D	D		D		
• Fast Moving Target	X	GP	GP								
• Slow Moving Target	X	GP	GP								
Fixed Target Track	X	GP	GP	D	X	D	D	D	D		D
Ship Detection and Track	X	X		D	D	D	D	D	D		D
Navigation Update	X	X	X	X	X	X	X	X	D		X
• Position	X	X		X	X	X	X	X	D		X
• Velocity	X	D	X	D	X	D	D	D	D		D
Terrain Avoidance	X	GP	D	X	X	X	X	X	D		X
Terrain Following	X	GP	D	X	X	X	X	X			D
Special Modes											
Beacon	X	X	X	X	X	X	X	X	X		X
ECCM	X	GP	X	GP	X	GP	GP	GP	GP		GP
TV			X								
SNIFF			X								

E = Existing.
GP = Growth potential (possible with present hardware).
D = Desired (requires new hardware).

Table 4-5. PRICE S2 OUTPUT CERs USED IN COST ANALYSIS

Resource	Complexity 0.8	Complexity 1.0	Complexity 1.2
	Integration = 0		
3.3	$9.47(I)^{1.007}$	$10.46(I)^{1.019}$	$11.61(I)^{1.029}$
3.5	$10.06(I)^{1.012}$	$11.02(I)^{1.025}$	$12.31(I)^{1.034}$
3.7	$10.64(I)^{1.017}$	$11.75(I)^{1.029}$	$13.16(I)^{1.038}$
	Integration = 0.5		
3.3	$12.31(I)^{0.990}$	$13.64(I)^{1.002}$	$15.21(I)^{1.012}$
3.5	$13.08(I)^{0.996}$	$14.43(I)^{1.008}$	$16.16(I)^{1.017}$
3.7	$13.88(I)^{1.001}$	$15.31(I)^{1.013}$	$17.30(I)^{1.021}$
	Integration = 0.7		
3.3	$13.97(I)^{0.987}$	$15.54(I)^{0.998}$	$17.41(I)^{1.007}$
3.5	$14.88(I)^{0.992}$	$16.49(I)^{1.004}$	$18.54(I)^{1.013}$
3.7	$15.81(I)^{0.996}$	$17.61(I)^{1.008}$	$19.85(I)^{1.016}$

Platform = 1.8 Utilization = 0.8 Application = 7.0

Table 4-6. OFP COST MODEL PARAMETERS

Dev. Alternatives	Parameter	F-16	FB/F-111 A/E/F	F-111D	B-52	F-4E	F-106
PIC	Complexity	1.2	1.2	1.2	1.2	1.2	1.0
	Resource	3.5	3.5	3.3	3.5	3.5	3.5
	Integration	0.5	0.5	0.3	0.5	0.5	0
	Add'l Int.	(3)	(3)	(3)	(3)	(2)	(1)
RIC	Complexity	1.2	1.2	1.2	1.2	1.0	0.8
	Resource	3.5	3.5	3.3	3.5	3.5	3.5
	Integration	0.5 & 0.7	0.7	0.5	0.7	0.5	0
	Add'l Int.	(3) & (5)	(5)	(5)	(5)	(4)	(2)
SIC	Complexity	1.2	1.2	1.2	1.2	1.0	0.8
	Resource	3.5	3.5	3.3	3.3	3.3	3.3
	Integration	0.7	0.7	0.5	0.7	0.5	0
	Add'l Int.	(5)	(5)	(5)	(5)	(4)	(2)

See Appendix F for parameter definitions

some of the integration and additional integration parameters. Under the RIC, we chose the 0.7 and (5) values for the F-16 OFP for Hughes and the 0.5 and (3) values for Westinghouse, since Westinghouse is the developer of the current F-16 radar.

Results of the development costs for the seven OFPs are shown in Figure 4-1. Totals range from $26.8M for the PIC to between $32.4 and 35.0M for the RIC, depending on whether Westinghouse or Hughes is the winner. The SIC costs total $33.9M. It should be noted that F-16 values reflect only the 25k program increase required by the addition of

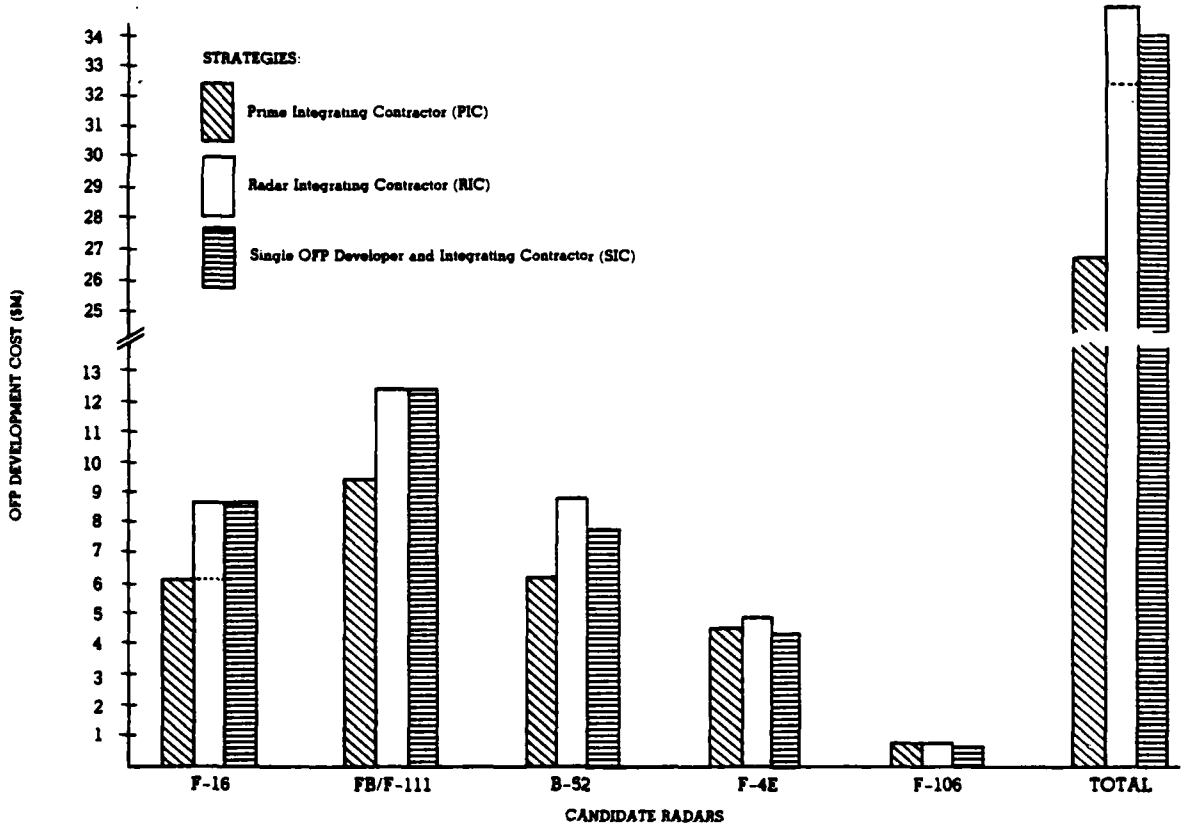


Figure 4-1. OFP DEVELOPMENT COSTS FOR THREE ACQUISITION STRATEGIES

new capabilities after the winner of the fly-off is declared. The cost for the 180k program is not included in Figure 4-1. The use of $5500/man-month equates approximately to FY80 dollars.

4.4.4 Sensitivity Analysis

We made a number of excursions to develop insight into the affects of various input parameters on software development cost. Westinghouse RIC parameters were used for the sensitivity excursions and only for the F-16 case, but the behavior of this case is representative of the total set.

4.4.4.1 Effect of Program Size on OFP Development Cost

Holding constant the parameters shown in Table 4-6 for the RIC, we varied the OFP size from 155k to 355k. The results of developing a program requiring 100 percent new code and 100 percent new design is shown in Figure 4-2. The relationship indicates that the cost of developing the F-16 advanced radar OFP (not included previously in Figure 4-1) would be over $22M using 180k instructions; an increase of approximately $1M would occur for each 10k of growth in the instructions input to the model.

An unresolved issue in the CMMR program is the size of the OFP required to achieve the capabilities demanded of the new radar. There is no doubt that the program software

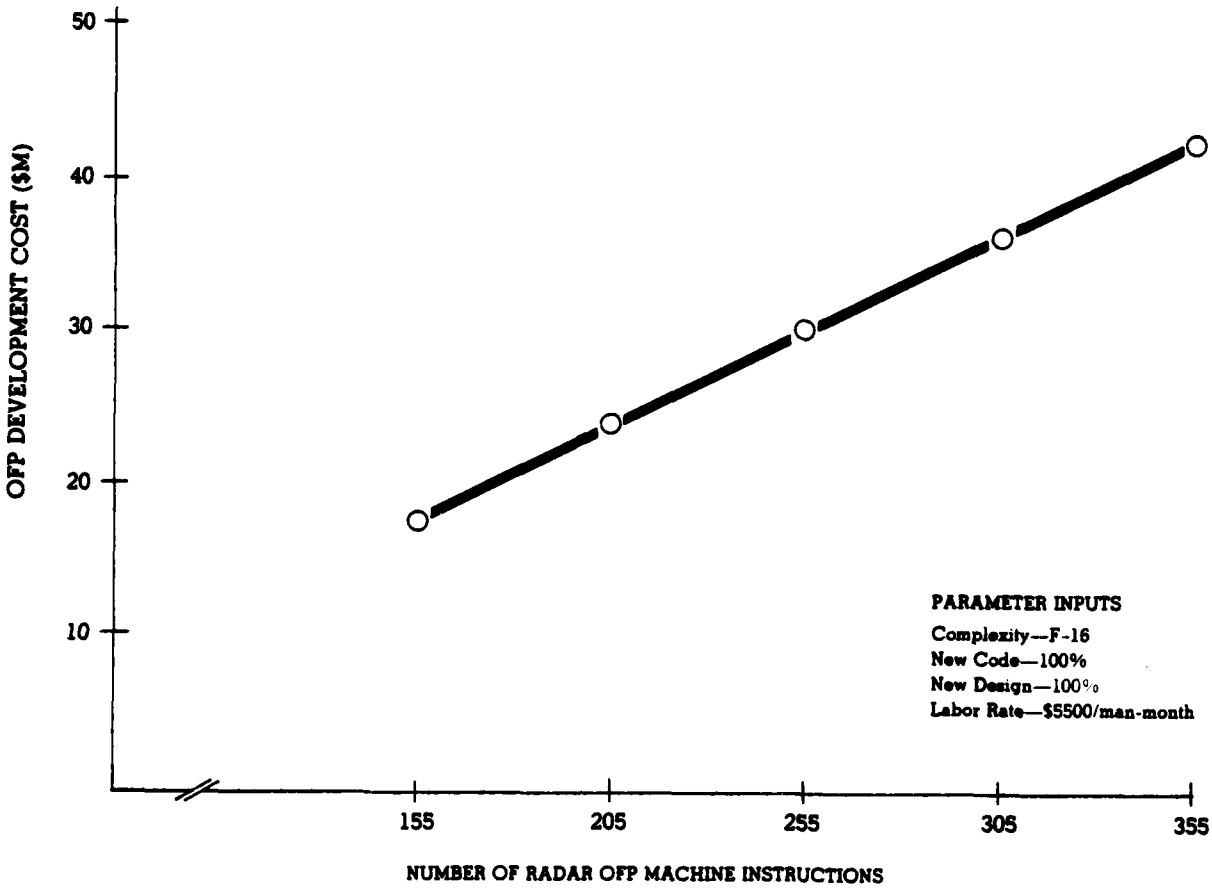


Figure 4-2. EFFECT OF PROGRAM SIZE ON OFP DEVELOPMENT COST
(Not including flight test and verification and validation)

development cost could be affected significantly by the extent to which the developer required more or less machine instructions to establish the CMMR capabilities.

4.4.4.2 Effect of Labor Rates and Code and Design Parameters on OFP Development Cost

Another troublesome factor in CMMR acquisition is the rapidly escalating cost of software labor. In this excursion, the F-16 parameters for RIC and a 180k program size were used again. Labor rates were varied from $4500 to $6500 per man-month. Three code and design parameters were assumed to gain insight into the leverage: 100 percent new code, 50 percent new design; 100 percent new code, 0 percent new design (highly unlikely); 50 percent new code, 50 percent new design.

The results, depicted in Figure 4-3, are linear and show that cost using the model is more sensitive to the amount of the new code than to the amount of new design. The CMMR Program Manager should be alert to changes in labor rate in developing his OFP estimates. The cost difference between $5000/man-month and $5500/man-month is $1.8M

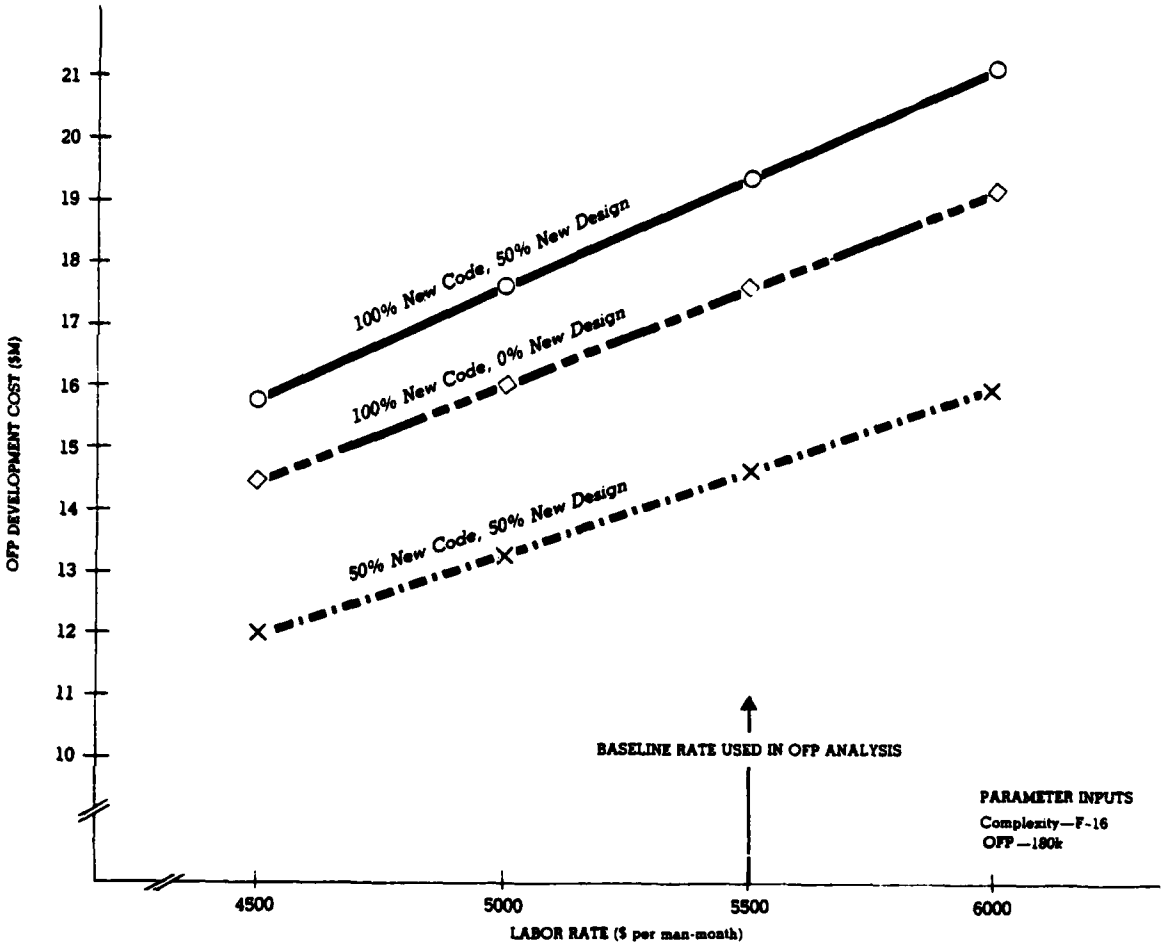


Figure 4-3. EFFECT OF LABOR RATE AND CODE AND DESIGN PARAMETERS ON DEVELOPMENT COST

for a 180k program with 100 percent new code and 50 percent new design. Furthermore, these parameters might be similar to those required to reprogram the CMMR OFP from assembly language to J73. These inputs result in a cost of $19.2M. A similar cost might be expected to recode the Hughes assembly language OFP.

4.4.4.3 Effect of Personnel Resources on OFP Development Cost

One of the concerns of the CMMR Program Manager in selecting the software acquisition strategy would certainly be the experience of the developer, as manifested in the degree of "learning" required to develop the CMMR software. In the ASD model, this aspect is considered quantitatively in the Resource parameter. Our baseline value was 3.5, which ASD considers as "average." We varied this input from 3.3 (above average) to 3.7 (below average). We also varied the new-code and new-design parameters from 20-percent new code, 10-percent new design to 40-percent new code and 20-percent new design, while holding the program size constant at 205k and the labor rate constant at $5500/manmonth.

The results are depicted in Figure 4-4, which shows two non-linear plots ranging from $16M to $9.5M for the 40-percent new code, 20-percent new design case and $13.5M to $8M for the 20-percent new code, 10-percent new design case. Although the average skill resource parameter (with some experience growth as discussed before) was used to develop the basic costs of Figure 4-1, the CMMR Program Manager must remain aware of the wide range of costs associated with the effort and capabilities of the personnel involved.

4.5 ANALYSIS OF SUPPORT COSTS AND HOL ISSUES

4.5.1 Estimate of Annual CMMR Baseline OFP Support Cost

As analyzed by LOGICON,* the software life-cycle can be defined in terms of three major phases: subsystem software development, weapon-system integration, and software maintenance. Each of these phases includes three general cost elements: OFP engineering, support hardware and software, and quality assurance. OFP engineering includes all technical manpower directly involved in designing, coding, testing, documenting, and managing an OFP. The maintenance phase includes two types of OFP engineering costs: corrective maintenance and modification maintenance. Corrective maintenance removes

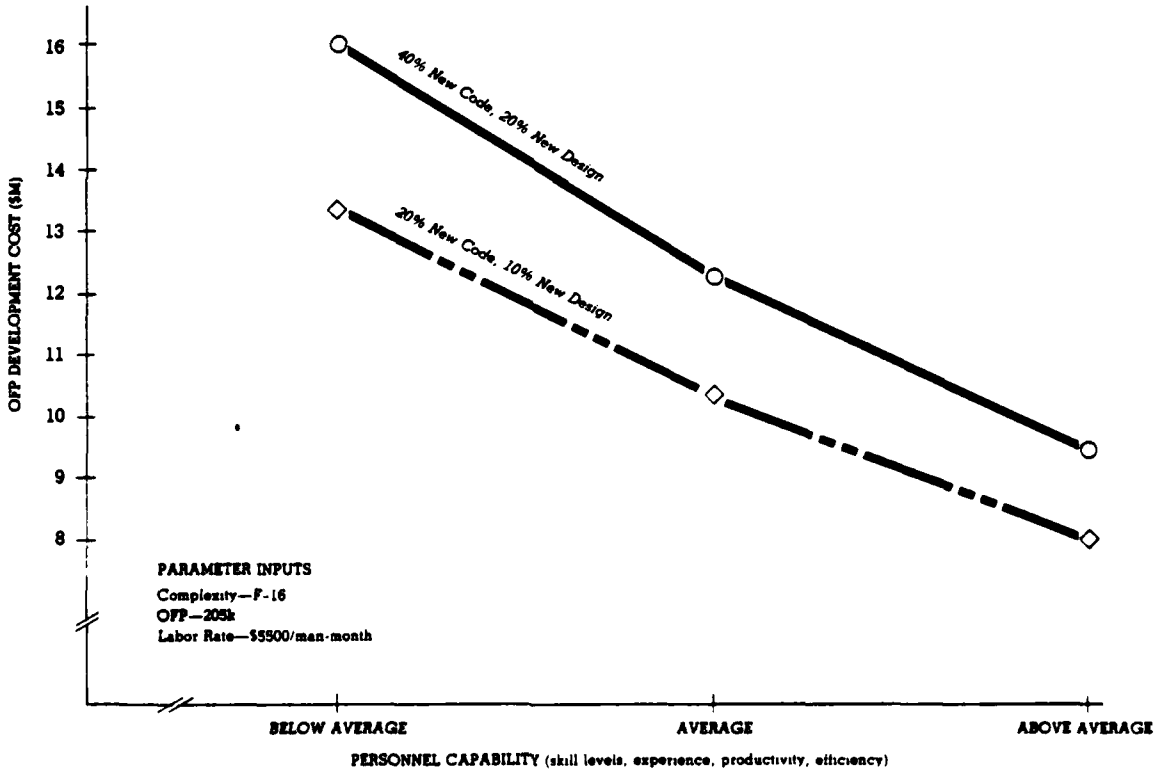


Figure 4-4. EFFECT OF PERSONNEL RESOURCES ON DEVELOPMENT COST

*LOGICON Report "Potential Effects of Standardization on Avionics Software Life-Cycle Cost," 29 June 1979.

latent errors from an OFP, while modification maintenance involves both adapting an OFP to accommodate changes in avionics hardware and enhancing an OFP to improve system performance. Support hardware and software cover the wherewithall for automatic-testing, aircrew-training, and direct OFP support. The latter includes: host-computer facilities, simulations, language translators, PROM loaders, ROM burners, etc. Quality assurance includes independent verification and validation and flight testing of the operational software.

Our review of other software cost analyses (see Appendix E) indicates that the cost of support ranges from 40 percent to 400 percent of the development cost. Systems that are threat-sensitive (such as electronic warfare systems) tend to fall at the high end of this range; systems with stabilized OFP requirements (such as flight control systems) tend to fall at the low end. A radar system falls in the higher end of the range.

The range of support alternatives for the CMMR software is exemplified on the one hand by the complex and costly Avionics Integrated Support Facilities at the ALCs, which were developed to support the more highly sophisticated weapons systems (B-52, F-16, F-15, F-111, etc.), and on the other hand by the centralized software support approach taken at Warner-Robins Air Logistics Center for all electronic warfare systems. As pointed out in Chapter Two, one of the initial steps the CMMR Program Manager must take is to form the CRWG which will, in turn, develop the CRISP. One of the key questions to be addressed by the CRWG is whether CMMR OFP support is to be centralized or distributed with the weapon system avionics.

For our support-cost analysis, we assumed that the F-16 205k radar OFP will be "baselined" after the second flight test program and will be maintained by the contractor for a three year period. The program then transitions to organic support with the contractor providing assistance between the third and fourth years. For the remaining six years, the Air Force supports the OFP organically.

The Resource parameter was changed from 3.5 to 3.3 over the initial three year period to reflect an assumed contractor experience in growth. At the fourth year, the Air Force experience is below average (3.7) but grows to 3.5 by year seven and to 3.3 by the 10th year. We inputted code-change values of 60 percent and 20 percent and design-change values of 30 percent and 10 percent in years 1, 3, 4, 7, and 10. The resulting non-linear curves in Figure 4-5 depict estimates of the annual support cost for the baseline CMMR OFP. They do not include costs for the other six OFPs, since these would vary considerably depending on the organic support concept selected (weapon systems vs centralized). Also, they do not include other costs discussed in this Section.

4.5.2 Effect of HOL Program Efficiency on OFP Development Cost

One of the most difficult assessments that will confront the CMMR Program Manager will be a judgment as to the maturity and modularity of the software design and code when the winner is selected after the fly-off. For Westinghouse, the modular radar OFP is planned to be written primarily in J73, ready to accept new code and new design for additional growth. For Hughes, the OFP would also be modular, but in assembly language.

To gain insight into some of the problems of reprogramming a previously designed assembly-language OFP into a HOL, we applied the RIC parameters to our assumed 205k OFP assembly code program at the end of the second flight test phase and developed new source code based on program efficiencies ranging from 100 percent to 50 percent. The

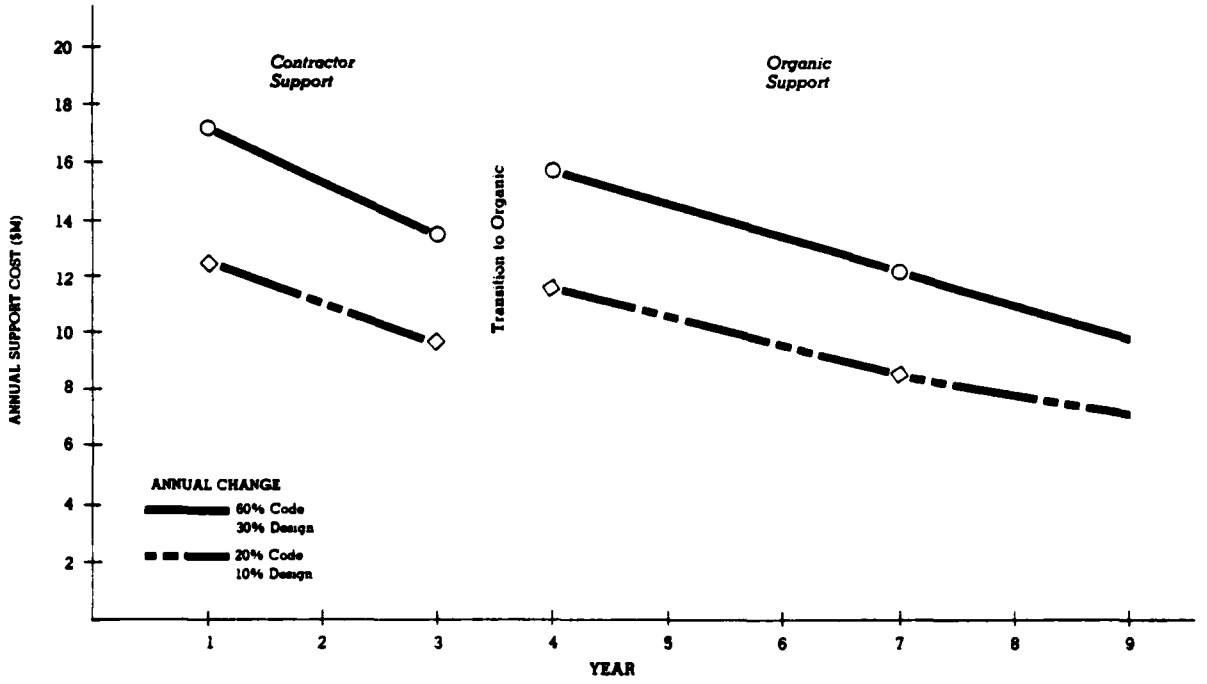


Figure 4-5. ESTIMATE OF ANNUAL F-16 OFP SUPPORT COST (10-year span)

resultant cost does not include any HOL support tools that may be needed. For Westinghouse, this might be valid, but for Hughes, at least a compiler targeted to the radar computer would be necessary unless this is developed and funded under the flight test program or a -1750A computer is used (for which compilers and other support tools are presumed to be available.)

For the purposes of this analysis, we defined a 100 percent efficient program as one that was previously written in assembly language source code, was entirely rewritten in a HOL, required 25 percent new design, and does not require additional memory due to recoding (highly unlikely). We did not attempt to analyze the problems associated with execution speed. As efficiencies change, the program size and timing will also change. For the purposes of this definition, inefficiency results from the compiler only.

Even though the ratio of HOL instructions to machine-language instructions can vary from 1:1 to 1:30, we used the average 1:4 conversion for J73, as suggested in the ASD model. We held the new design at 25 percent and varied the new code's efficiency from 1.0 to 0.5. This also "grew" the program from 205k to 308k.

The results, depicted in Figure 4-6, show a non-linear spread of additional OFP development cost ranging from 0 to $18.5M for a variation of efficiency from 100 percent to 50 percent. The numbers would have been higher if the 25 percent new design had increased.

Rome Air Development Center (RADC) has suggested that an efficiency range of 80-85 percent might be anticipated for the J73 compiler; the more pessimistic figure represents a penalty of approximately $7M under the stated assumptions.

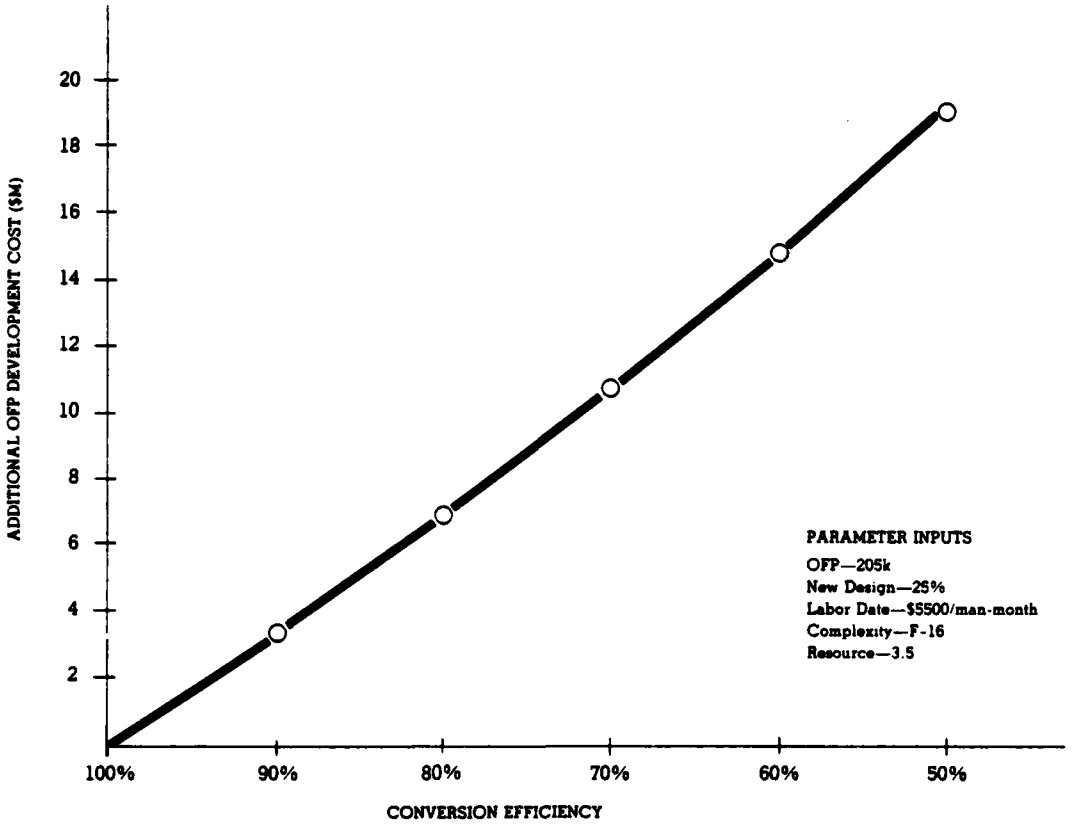


Figure 4-6. EFFECT ON DEVELOPMENT COSTS OF CONVERTING ASSEMBLY LANGUAGE TO HOL

Use of the ASD model provides an estimate of $23.9M for the cost of developing a *new* 205k HOL OFP with 100 percent new code and 100 percent new design. We do not feel, however, that it is reasonable to achieve a 100 percent new-code HOL implementation of the radar OFP. Therefore, setting 80 percent as a more realistic achievement, the penalty due to compiler inefficiency alone is approximately $5.6M.

Any penalty in development cost due to HOL conversion may be offset by a savings in support cost over the long run. As an indication of the order of magnitude of these savings, we used an average yearly support cost of $9.5M (see the lower curve in Figure 4-5) and applied it for a ten-year period. The total value—$95M—represents a yearly change of 20 percent new code and 10 percent new design in OFP alone, under the assumptions given. The ACCX Report on the cost model indicates that approximately 20 percent of development cost occurs in the coding process. Furthermore, Hughes* has stated in their LCC model that 20 percent of the software support cost can be attributed to coding, indicating that OFP costs for changes during support are similar to those incurred during new development. Thus, the approximate projected support cost for coding alone is

*Source: Hughes Report: "Predictive Software Cost Model", Vol. I, June 1980.

(0.2) (95) = $19M. RADC has indicated that programmer productivity (number of lines of code per month) could nearly double when a HOL rather than assembly language is used. Consequently, the potential support cost saving resulting from use of a program written in 80-percent HOL (assuming the same ratio applies to the annual 20 percent new code) is (0.8) (0.5) (19) = $7.6M for the ten-year support phase. Even if our support cost is somewhat overestimated, the approximate $5.6M investment penalty for compiler inefficiency would be paid back over a ten-year period. Not considered in the investment penalty are requirements for increased memory hardware if efficiencies are much less than the RADC range. However, memory is becoming an increasingly minor cost driver.

The above observations are drawn from a very specific, but reasonable set of assumptions concerning just one OFP (the F-16) in the CMMR program. Since the CMMR support concept has not been finalized, we do not believe that they make a convincing case for use of HOL; however, neither do they provide compelling assumptions to apply for exemption from HOL. Rather, they indicate the potential for longer term support benefits, particularly given the possible application of a HOL in other areas of the architecture.

4.5.3 Assembly Language Translation to J73

For this analysis, we assumed that the present 180k Hughes OFP grew to 205k as before. We also assumed that the RDP portion (75k) of the OFP is rewritten in 80-percent HOL, the PSP portion (120K) is rewritten in 50-percent HOL and the "Misc" portion (10k) remains written in assembly language source code. Again, we used the $5500/man-month, the 100-percent new code, and 25-percent new design parameters.

The results are a linear plot of program size vs cost, which varies by 24k and $2.5M respectively for each 20 percent of inefficiency. For example, with a 80 percent HOL efficiency for the assumed assembly language OFP, the size of the Hughes program is 24k more (229k) and shows a cost growth of $2.5M. Table 4-7 illustrates this program. The $2.5M does not include potential programmer man-hour saving resulting from HOL coding or additional costs for compiler development etc., as discussed before.

Table 4-7. EFFICIENCY

	205k OFP (Ass. Lang.)	205k HOL/MIX OFP (100% Efficiency)	229k HOL/MIX OFP (80% Efficiency)
RDP	75k	60k HOL (80%) 15k AL	72k 15k
PSP	120k	60k HOL (50%) 60k AL	72k 60k
MISC	10k	10k AL	10k
	205k	205k	229k

*LOGICON Report, 29 June, 1979

The CMMR Program Manager will need to monitor continually the problems and cost variances associated with using a HOL—especially if the program initially exists in assembly language and rewriting of its source code is required. For this excursion, we only analyzed the growth in memory size due to compiler inefficiency, but there is also a major concern associated with program execution speed: a thorough review of the present Hughes OFP will be required if Hughes is the winner and the radar OFP is to be rewritten in J73.

4.6 RESULTS

Table 4-8 ranks the three CMMR software acquisition strategies defined in Section 4.2 according to the criteria identified in Section 4.3. The rankings are at this time necessarily subjective. It is important to keep in mind that our judgments are based on the assumption that the CMMR market is sufficiently large to sustain more than one radar manufacturer. If this assumption is not valid, the difference between RIC and SIC is almost indistinguishable.

For the Operational Capability criterion we have ranked the SIC alternative as the most attractive. The contractors for all three approaches could meet the radar OFP acquisition needs, given sufficient funding and time. However, we feel that the SIC would be in a better position to control and maintain the configuration of seven OFPs, build growth and flexibility in the OFP design, and accommodate threat changes to the radar OFP in a timely manner.

For the Cost criterion, the PIC is the most attractive alternative based on the results of our analysis of acquisition costs. The RIC and SIC are ranked equally, although the RIC

Table 4-8. RANKING OF CMMR SOFTWARE ACQUISITION STRATEGIES

Ranking Criteria	PIC	RIC	SIC
Operational Capability (Threat Accommodation)	−	0	+
Cost			
Acquisition	+	0	0
Initial Support	−	0	+
Schedule			
Radar OFP	−	+	0
Weapon System IOC	+	−	0
Risk	+	−	0
Supportability	0	−	+
Management (Ability to control costs and flexibility to accommodate more than one radar manufacturer)	+	−	0

\+ Most Attractive
0 Moderately Attractive
− Least Attractive

cost could be lower or higher than the SIC cost, depending on the winner of the fly-off. For the initial support cost, SIC is the most attractive because of the contractor's "learning" experience.

We have also split the Schedule criterion into two parts. The RIC alternative is the most attractive for meeting the radar OFP schedule because of the contractor's unique experience with the radar hardware and software. For the weapon system IOC schedule, the PIC alternative is the most attractive because of the individual contractor's knowledge of the avionics software, his responsibility for the radar OFP, and his overall responsibility in meeting the weapon system IOC. Also, since there would be five PICs—one for each of the five candidate aircraft— they could work in parallel.

For the Risk criterion, the PIC alternative is the most attractive because of the difficulty of integrating at least five of the radar OFPs into the rest of the complex avionics software suites. This ranking also recognizes the possibility that a second radar manufacturer would be required if the market size is large enough, compounding the risk with the RIC strategy.

For the Supportability criterion, we have ranked the SIC alternative as the most attractive. The SIC develops an overall understanding of each of the candidate aircraft's weapon system requirements and how the radar OFP is to meet these requirements. The SIC would "grow" as the integration problems encountered in each aircraft are understood and resolved; he would be able to maintain an unbiased perspective with regard to both the avionics and radar software. The SIC would be in a better position to assist the Air Force in establishing organic CMMR software support because his responsibilities are exclusive of the radar manufacturer's.

The PIC alternative is ranked as the most attractive for the Management criterion, because less government management, engineering, and contracting resources would be required. For example, the SIC alternative introduces "third party involvement" in the problems of radar software development and integration.

All three alternatives would undoubtedly require both prime and subcontract arrangements.

CHAPTER FIVE

IMPLEMENTATION ACTIVITIES

5.1 ACQUISITION STRATEGY

The relative importance of the evaluation criteria described in this report is dependent on Air Force priorities. The apparent thrust of the current government deliberations with respect to CMMR is that, within reason, schedule considerations will defer to cost considerations. The dual competitive fly-off appears to represent the emerging overall procurement philosophy. Given this philosophy, the assumption that two or more aircraft are involved, and the willingness of the Air Force to accept considerably more technical and management involvement in the program than is normally the case, the following overall acquisition strategy is envisioned for the total (hardware and software) CMMR program:

1. Use of a single OFP developer for the software portions of CMMR. If there is a market requirement for more than one radar hardware supplier, the SIC strategy appears to be the preferable approach for the OFP development. Otherwise RIC and SIC appear equally attractive.
2. Single design of the two common major LRUs: Programmable Signal Processor (PSP) and Radar Computer.
3. Use of MIL-STD-1750A ISA and MIL-STD-1553B for at least the Radar Computer. More information is needed on the PSP requirements before the ISA can be established for this LRU.
4. Other possible hardware GFE includes most SRUs of the transmitter, receiver-exciter, antenna and servo, associated controls and displays, and remote terminal units for the multiplex bus installation. The extent to which these units are supplied as GFE is dependent on the target aircraft.
5. Use of a growth-oriented software implementation approach. This implies a structured programming approach following the philosophy, if not the precise requirements, of MIL-STD-1679. Another approach might be to use Ada as a design language to achieve the same objective.
6. Transfer of advanced algorithms, such as NCTR, to facilitate growth from laboratory programs (e.g. PE 63742) at a later time; sponsorship of other algorithms not undertaken by the laboratories to avoid concurrent development expense for each aircraft system.
7. Use of the J73 language for those portions of the architecture that are susceptible to change (e.g. threat-sensitive) and for which execution time can be accommodated. J73 should not be applied indiscriminately for subroutines that are standard in software libraries and that are reasonably stable (e.g. Kalman Filter processing algorithms). However, it is expected that, with time, more and more of these subroutines will be written in a HOL.

8. Use of in-house resources and personnel when possible. For example, SEAFAC could be used for independent verification and validation, if MIL-STD-1750A is directed.

9. Use of GFE compilers for development and support. Initially, this may require the use of GFE machines (e.g. DEC-10) as well. Eventually the Federal Software Exchange Center will provide more portable tools.

10. Reliance on the contractor for initial support of both hardware and software to permit more considered examination of the potential long-term alternatives.

5.2 EARLY IMPLEMENTATION ACTIVITIES

Regardless of the ultimate selection of strategy, there are implementation activities that should be instituted by the Program Manager. These are depicted in Figure 5-1 along with their appropriate order of occurrence.

The key actions that are required on the part of Headquarters USAF are (*refer to* Figure 5-1 for activity references): Activity 1—the selection of the candidate aircraft and allocation of funds and Activity 2—the issuance of the PMD. This formal direction will be reflected in the ASD Program Management Plan (PMP) together with the acquisition strategy (Activity 3). Some adjustments to the acquisition strategy will be required to account for the Headquarters USAF direction.

We believe that the most critical initial step following the submittal of the PMP is a definitive operational requirements audit (Activity 4). There is no formal documentation to establish priorities for each mode in each potential aircraft; the users must be surveyed to establish such priorities. This information would permit the initiation of three parallel activities: the formation of a Test and Evaluation Working Group (TEWG) (Activity 5); the formation of an Interface Control Working Group (ICWG) (Activity 10); and the conduct of a series of LCC tradeoffs (Activity 7).

Activities 5 and 6 define operational suitability and permit the TEWG to coordinate and develop a Test and Evaluation Master Plan. An important part of this plan concerns the careful structuring of an "apples to apples" fly-off and preparation of a condensed Development Test/Operational Test Plan thereafter.

5.3 FOLLOW-ON ACTIVITIES

Activity 7 in Figure 5-1 indicates LCC tradeoffs that should be conducted to determine the benefits of including or excluding CMMR common components for each candidate aircraft. The resulting cost values should be used to establish inputs for the POM and FYDP, and to scope the SOW for the procurements (Activities 8 and 9).

Activities 10, 11, and 12 provide a forum for considering the physical and operational envelopes established by each aircraft candidate and conducting the associated partitioning trade-offs. The results can be used to revise the current preliminary specifications and to establish ICDs. The implied software processing partitioning is used as an input to the CRWG.

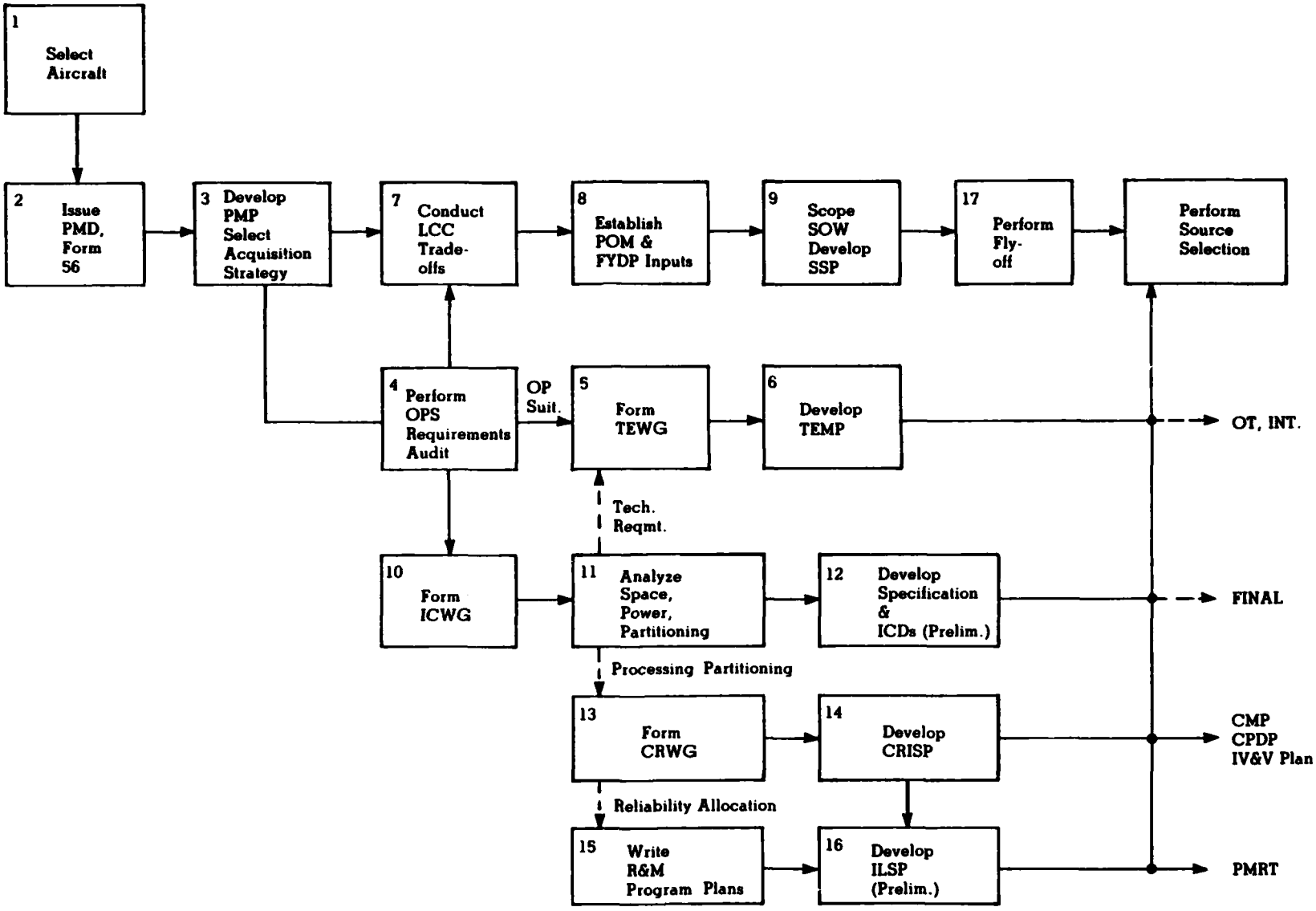


Figure 5-1. IMPLEMENTATION STEPS FOR CMMR ACQUISITION STRATEGY

Activities 13 and 14 develop alternatives for the computer resources and assist in the development of the preliminary CRISP. This series of activities culminates in the issuance of one CMP and approval of the CPDP and independent verification and validation plan.

Activities 15 and 16, through space and power partitioning, establish the overall physical parameters needed for a Reliability and Maintainability Program Plan. Their output provides a basis for the Integrated Logistic Support Plan.

CHAPTER SIX

CONCLUSIONS AND RECOMMENDATIONS

6.1 INTRODUCTION

This report has been structured to help Air Force planners make the decisions that are required to determine the future of the CMMR program. If the program is given the go-ahead, it will be one of the Air Force's most ambitious undertakings in terms of technical complexity. It is the first attempt to develop common radar hardware and software for extensive fleet retrofit. Because of the large initial investment required and the disparate technology, production, and modification programs affected, it is apparent that consolidation will not be an easy task.

6.2 CONCLUSIONS

The following general conclusions have been reached by ARINC Research.

6.2.1 Review of Software Guidelines, Policies, and Standards

- The thrust of current guidelines applicable to USAF digital avionics systems' acquisition is toward standardization of data interfaces, use of digital architecture, and use of HOLs. The major Standards with potential for significant impact on the CMMR program are MIL-STD-1553B (Mux Bus), -1750A (ISA), and -1589A (J73 HOL). MIL-STD-1760 (Aircraft Stores/Electrical Interface) is in draft form and may be applied to CMMR.
- Since CMMR is basically a standardization initiative that is highly software intensive, it is reasonable to project that all the Standards mentioned above will be applied to the program unless compelling reasons for exemption can be shown. Our analysis does not reveal any such compelling reasons. Rather, use of the Standards could provide long-term benefits, even if only partially applied.
- Software development requires the same development process and controls as hardware development. Air Force Regulation 800-14, Volumes I and II, provides program management guidance for the acquisition and support of computer resources in systems; it should be followed closely.

6.2.2 Review of Technology Programs

- Our review of nine radar technology programs for applicability to CMMR software

development revealed the following four that will have significant impact on CMMR:

Program Title	PE/Project	Status	Impact on CMMR
RPSP	64201/2519	Continuing	Software algorithms transfer to F-16 if done ASAP
NCTR	63742/1177	Continuing	Algorithms transfer to CMMR
ERIP	64201/2259	Cancelled	TF radar portions applicable to CMMR
EAR	63241/1206	Project Complete	Technology Source for some algorithms and software

- The IDA program establishes and promotes standards for digital avionics architecture but does not enforce their use. Existing facilities available to the Program Manager to provide the capability to implement IDA standards include the ASD/ENASD SEAFAC, the ASD Computer Center, and the AFWAL/AA AVSAIL simulation facility.
- Although the MATE program will not complete FSED until 1985, it could be directed for application to CMMR with use of interim contractor support until MATE hardware is available.

6.2.3 Production Program Review

- The radar hardware and software designs for all three aircraft examined (F-15, F-16, F/A-18) are still being updated.
- Radar memory requirements for all three aircraft have increased by a factor of between 3 and 6 due to deficiencies in design and new software requirements.
- Both the avionics and radar software for all three aircraft have been designed for modularity. The modular software design should enhance software maintenance.
- The F-16 and F/A-18 aircraft employ some HOL source coding in their main computer OFPs. However, the F/A-18 APG-65 radar OFP is written totally in assembly language and there are no plans to convert it to a HOL. The F-16 Advanced Radar OFP is planned to be written in J73.
- Software support for the radar OFP for all three aircraft is accomplished under a prime/subcontractor arrangement at present. The Air Force as yet has not committed itself to organic support.

6.2.4 Acquisition Strategies

- The preferred software acquisition strategy implied by the ranking of our alternatives is a single OFP developer and integrating contractor (SIC).
- The relative importance of the evaluation criteria developed in this report are dependent on Air Force priorities. The apparent thrust of the current government deliberations with respect to CMMR is that, within reason, schedule considerations will defer to cost considerations. The dual competitive fly-off approach appears to be the emerging overall procurement philosophy. Given this philosophy, the assumption that two or more aircraft are involved, and the willingness of the Air Force to accept considerably more technical and management involvement in the program than is normally the case, the following overall acquisition strategy is implied for the total (hardware and software) CMMR program:
  - Use of a single OFP developer for the software portions of CMMR. If there is a market requirement for more than one radar hardware supplier, the SIC strategy appears to be the preferable approach for the OFP development. Otherwise RIC and SIC appear equally attractive.
  - Single design of the two common major LRUs: Programmable Signal Processor (PSP) and Radar Computer.
  - Use of MIL-STD-1750A ISA and MIL-STD-1553B for at least the Radar Computer. More information is needed on the PSP requirements before the ISA can be established for this LRU.
  - Other possible hardware GFE includes most SRUs of the transmitter, receiver-exciter, antenna and servo, associated controls and displays, and remote terminal units for the multiplex bus installation. The extent to which these units are supplied as GFE is dependent on the target aircraft.
  - A growth-oriented software implementation approach. This implies a structured programming approach following the philosophy, if not the precise requirements, of MIL-STD-1679. Another approach might be to use Ada as a design language to achieve the same objective.
  - Transfer of advanced algorithms, such as NCTR, to facilitate growth from laboratory programs (e.g. PE 63742) at a later time; sponsorship of other algorithms not undertaken by the laboratories to avoid concurrent development expense for each aircraft system.
  - Use of the J73 language for those portions of the architecture that are susceptible to change (e.g. threat-sensitive) and for which execution time can be accommodated. J73 should not be applied indiscriminately for subroutines that are standard in software libraries and that are reasonably stable (e.g. Kalman Filter processing algorithms). However, it is expected that, with time, more and more of these subroutines will be written in a HOL.
  - Use of in-house resources and personnel when possible. For example, SEAFAC could be used for independent verification and validation, if MIL-STD-1750A is directed.
  - Use of GFE compilers for development and support. Initially, this may require the use of GFE machines (e.g. DEC-10) as well. Eventually the Federal Software Exchange Center will provide more portable tools.

- Reliance on the contractor for initial support of both hardware and software to permit more considered examination of the potential long-term alternatives.

6.2.5 Implementation Activities

- The three key actions in the formal direction required from Headquarters USAF are (1) selection of the candidate aircraft, (2) allocation of funds, and (3) issuance of the PMD.
- The formal direction from Headquarters USAF should be reflected in the ASD Program Management Plan (PMP).
- The most critical initial step following the submittal of the PMP is a definitive operational requirements audit, followed by (1) formation of a Test and Evaluation Working Group (TEWG), (2) formation of an Interface Control Working Group (ICWG), and (3) conduct of a series of LCC tradeoffs.

6.3 RECOMMENDATIONS

ARINC Research recommends that Air Force planners responsible for the CMMR effort take the following actions:

- Use a single OFP developer and integrating contractor for the CMMR software acquisition strategy.
- Develop a detailed activity road map to implement the acquisition strategy.

APPENDIX A

REFERENCE MATERIAL

Department of the Air Force, Aeronautical Systems Division, *Airborne Systems Software Acquisition Engineering Guidebook* (first six guides). NTIS accession; Nov. 77-Nov. 78.

_____, Aeronautical Systems Division, *Management Guide to Avionics Software Acquisition*. 4 Vols, NTIS accession; June 1976.

_____, Air Force Avionics Laboratory, Interim Technical Report, *Electronic Warfare Software*. Second printing, 20 December 1979.

_____, Air Force Test and Evaluation Center, "Software OT&E Guidelines," *Software Maintainability Evaluation Handbook*. Vol. II, August 1979.

_____, Aeronautical Systems Division, Deputy for Avionics Control, *Final Report on B-52 OAS HOL Selection and Future JOVIAL HOL Standardization/ Control*; 24 July 1978-4 August 1978.

_____, Air Force Avionics Laboratory, *Predictive Software Cost Model Study*. Vols. I and II, June 1980.

_____, Aeronautical Systems Division, Strategic Systems SPO, *B-52 Offensive Avionics System Computer Resources Integrated Support Plan*, 27 August 1979.

_____, Aeronautical Systems Division, F-16 SPO, *F-16 Multinational Computer Resources Integrated Support Plan*. Vol. 0, Revision 1, 16 December 1978; Vol. I, Revision 2, 2 February 1979.

_____, Aeronautical Systems Division, F-15 SPO, *F-15 Avionics Computer Resources Integrated Support Plan*. Revised March 1978.

_____, Aeronautical Systems Division, Comptroller, Directorate of Cost Analysis, *Avionics Software Costing, A Derivative Approach*, January, 1980, and *TI-59 Handheld Calculator Software Cost Estimating Model, User's Handbook*; Vol. I, March 1980 and Vol. II, June 1980.

Final Report of the Joint Logistics Commanders' Software Workshop. Vol. 1, 1 October 1979.

Standardization in Military Avionics Systems Architecture, an IEEE/ AES-Seminar, Dayton, Ohio, 28 November 1979.

Proceedings of the IEEE National Aerospace and Electronics Conference, NAECON, 3 Vols, 15-17 May 1979 and 20-22 May 1980.

Proceedings of the Professional Societies of the Naval Air Station Patuxent River Complex, *The Profound Impact of Airborne Software,* 21 January 1980.

APPENDIX B

POINTS OF CONTACT

SOFTWARE

Organization	Name	Functional Area
DOD		
	Mark Grove	HOL/ISA Standardization
	Dave Fisher	Ada HOL
HDQRS USAF		
RDPV	Maj. Doc Dougherty	Avionics Software
RDPV	L/C Ron Radford	CMMR Direction
RDM	Leyton Lomas	AFR 800-14
HDQRS AFSC		
XRF	Maj. Al Kopp	A.F. J-73 Designated Control Agent
XRF	Maj. Larry Fry	JLC Software Mgmt. Subgroup
SDDL	Clark Walker	ATLAS Designated Control Agent (AF)
SDDS	Maj. Phil Merkley	Configuration Management
SDNA	Capt. George Radic	CMMR Direction
ASD		
XRE	Mike Yackowski	Avionics Software
EN	Dr. Dick Sylvester	ASD Weapon System Computer Resources
XRE	Jerry Duchene	IDA
AEG	LTC Danhof	MATE
AX	Ajmel Dulai	1750A/1553B Control Agent
RW	Maj. Carl Canter	CMMR Program Manager
ENASD	Capt. Rick Pensworth	SEAFAC-1553 Control Agent-1750 Control Facility
AEG	Oscar Sepp	ATLAS Language Control Agent
AWZ	Charles Marshall	Configuration Mgmt. of Computer Resources
ACCX	Capt. Bob Gaffney	ASD Cost Analyst
RADC		
ISI	Richard Nelson	Information Processing Branch Chief
ISIS	Dick Slavinski	J73 LCF/Language Control Agent
ISIS/L	Lorraine Duvall	Design/Analysis Center for Software (DACS)

AFWAL		
AA	Dr. Don Moon	J73 Compiler
AA	Frank Scarpino	J73 Compiler
HDQRS AFLC		
LOEC	Harry Mason	ATLAS Control Agent
LOWW	Ray Armstrong	CMMR Direction
NAVMAT		
042	Richard Berry	Joint Configuration Mgmt. Reg.
08Y	Owen McOmber	Navy HOL Control Agent
08Y	Rick Harrington	NAVY ISA Control Agent
NAVAIR		
533	Cal Showalter	TECR Program Manager
NAVSEA		
408	Denise Johnson	CMS-2M Control Agent

WEAPON SYSTEM

		Area
F-15		
ASD/TAFA	Bob Hill	Avionics Software
TAFEA	Nate Pringle	Radar Hardware
TAFEA	Larry Lang	Radar Software
WRALC/MMEC	Charles Singleton	Avionics and Support Software
MMECA	Bob Anderson	OFP Support
MMIRL	John Foxbower	Radar Item Manager
F-16		
ASD/YPA	Ron Embertson	Avionics Software and Support
YPR	Maj. John Warner	Radar Hardware
YPEA	Lt. Brian Kawol	FCC Software
YPEA	Charles Grooms	Radar Software
YPR	Maj. Ron Fuchs	Advanced Radar HW/SW
OOALC/MMARE	Dean Gordon	Avionics Support
MMARE	Wayne Bates	Avionics and Support Software
MMARE	David Sanders	Radar Hardware
MMECA	Dave Ericson	Radar Software
MMECA	Floyd Bergout	FCC Software
F-18		
PMA 265	John Hook	Avionics
533	Chuck Ogelsby	Avionics Software
549	Bill Seegars	Radar Hardware
533	Chuck Ogelsby	Radar Software
522	Larry Bianco	Avionics Software Support
549	Don Jackson	Avionics Support

APPENDIX C

F-15, F-16, F-18 PROGRAM CHECKLIST

WEAPON SYSTEM/SUBSYSTEM REQUIREMENTS:

Aircraft Mission (primarily A/A or A/G), Functions, Weapons (Missiles, Guns, Bombs) Major Subsystems

Radar Requirements—How Identified/Specified

Other Uses for equipment

DIRECTION:

PMD, Form 56 Status

Current policies and procedures for avionics/software—Listed in Direction

Dev/Prod vs. support—To different organizations

MANAGEMENT

Program Management Plan—Separate management for radar

Separate organizational management relationships

OFP Management vs. Subsystem (radar) Management

Advanced Procurement Plan—Prime/sub relationships (contracturally binding for radar)

Radar vs weapons system—Change procedures/categories ECPs, configuration Management procedures, CCB (CMP), approval authorities for change

List all contractors involved in hardware/software development and Production; list their relationships

Management problems/lessons learned/method for problem tracking

SCHEDULE:

Weapon system IOC/update for mission avionics

Radar separate from weapon system
First Production Aircraft—with Radar Delivery

PMRT Date

Technology inputs—Any required/planned

Software/hardware updates—planned/approved—Block changes—When

DEV/PROD/SUPPORT PHILOSOPHY (OVERALL):

Design-standardize hardware and/or software Designed for changing requirements

BIT vs ATE Concept, % BIT. Was trade-off study performed

F^3 vs performance specs for avionics/radar

% radar hardware "shared" with other avionics

Test philosophy—independent verification and validation

Support philosophy—RIW, organic, ICS, field, depot, factory

Responsible ALC/Navy support facility

CONTRACTUAL REQUIREMENTS

Prime/Sub-Relationships

CDRL documents (CRISPS, CPDP, CPCI'S, etc.)—format

Any restrictions on data

Documentation adequate—too much

Contract incentives/warranties/guarantees

Reliability/improvement efforts

Specs and Standards Directed (Language Specified)—How Controlled

Hardware architecture specified. (Avionics/Radar)

Hardware specified with software or separate

Processor specified

I/O (Interface) Requirements/Specifications/Standards

Parts qualification procedures—software quality—how handled

Second-source requirements

Support requirements

Training requirements

Method used to motivate quality design

OPERATIONAL SOFTWARE DEVELOPMENT AND PRODUCTION
(Leading to Software OFP(s))

Developer

Type architecture —Hierarchical/distributed/structured programming/ top-down design-Directed

Control of development/production

Functional flow diagrams—Block diagrams

Language (assembly—HOL—both); processor used

Development cycle—Part I Spec complete—Update

Functional spec (flow diagram)

Overall memory requirements

- % PROM, ROM, EPROM—How changed (SE) and where
- % hardware/firmware
- Number of words—word length—instructions
- Speed—IPS—capacity required/ cost per mode/duty cycle
- Semi-conductor vs magnetic core vs disc
- Capacity filled —Provisions for growth

Hardware, model number/designation

Baseline established—When—If so, spec number

Latest revsision/status

Design reviews—How often

Modular design concept—Functional (A/A, A/G, etc.)

- Major Modules—Designed for change (Is most difficult last —One function doesn't affect other programs)

Portability considered

Number of OFP's; which Version

- Radar/CC/Other
- Functions/Language/Size—Source codes objective—Number of Lines

Documentation adequate—Too much, Standardized

Technical problems/Lessons learned

SUPPORT SOFTWARE

Number of CRISPS

Number of personnel involved/Government/Manufacturer

Maintenance concept

Number of support programs—Functions/Languages

- Compiler(s)—Languages, host computers, target computer(s)
- Assemblers/cross-assemblers
- Emulators
- Simulators
- Comparators
- Test Diagnostics
- Program Library

EPROM reprogrammer status/cost

AISF/AIS—Differences

ATE Language

Portability Considered

TESTING

V & V "agent" for software

Check-out status

TRAINER SOFTWARE

Operational flight trainer

Mission trainer

Cockpit procedures trainer

Weapon system trainer

APPENDIX D

WEAPON SYSTEM ARCHITECTURES

1. INTRODUCTION

This appendix provides a general background of the F-15, F-16, and F/A-18 avionics suites.

2.0 F-15 AVIONICS ARCHITECTURE

2.1 Weapon System Overview

The F-15 "Eagle" is a twin-turbofan, single seat, air superiority, tactical fighter manufactured by McDonnell Aircraft Corp. (MCAIR). The U.S. Air Force's total planned buy is 729 aircraft, consisting of 350 A models (basic single seat), 54 B models (two-seat trainer), and 325 C and D models (single and two-seat aircraft equipped with modification to the fuel tank system for increased range.) With the delivery of aircraft No. 404 in June 1979, the Air Force began procurement of the F-15 C and D models. Current production plans indicate the last of the 729 U.S. models will be delivered in November 1983, although the total quantity is still an issue.

The F-15 has five weapon stations that permit carrying of up to 15,000 pounds of ordnance. Its armament consists of one M61 20mm cannon, four AIM-7F Sparrows, and four AIM-9L Sidewinders. The new Advanced Medium Range Air-Air Missile (AMRAAM) is slated to replace the AIM-7F. The F-15 is scheduled to transition to Warner-Robins Air Logistics Command (WR-ALC) in October 82.

2.2 Avionics Architecture and Hardware

The F-15 avionics suite has been designed as an integrated system. A central computer (CC) primarily performs mission oriented calculations. The characteristics of the IBM AP-1 CC are listed in Table D-1.

The F-15 digital integrated avionics system architecture is shown in Figure D-1.All interface with the CC is digital and all peripherals, except two, are connected to NON-1553 redundant multiplex buses consisting of two identical transmission lines. Each transmission line is a shielded twisted wire pair, providing two paths for digital data transmission. One transmission line carries the data and the other line is for timing. Data transmission has a word serial, bit serial, time division multiplex format. This means that data is transferred one bit at a time and the line is time-shared between the CC and its peripherals.

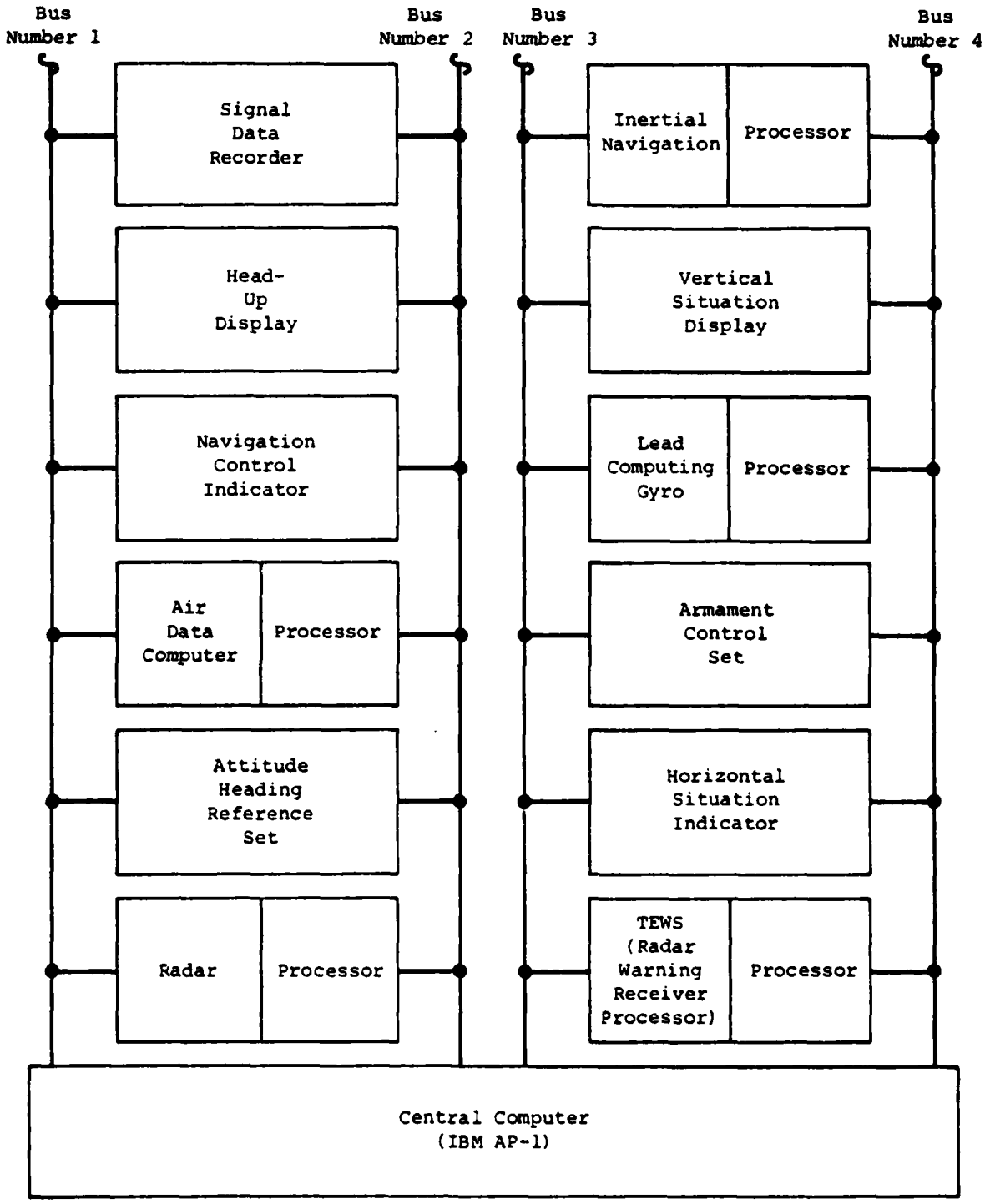


Figure D-1. F-15 AVIONICS ARCHITECTURE

Table D-1. CHARACTERISTICS OF F-15 CENTRAL COMPUTER (IBM AP-1)

- 16,384, 34 bit (2 parity), words of memory storage, expandable to 24, 576 words.
- Capable of interfacing with 4 channels and up to 15 peripherals per channel.
- Capable of performing 340,000 instructions per second.
- Data word length—half word is 16 bits and full word is 32 bits.
- Requires no adjustments in aircraft.
- No memory loss during power transients.
- Full parallel operation.
- Uses fixed point, two's complement, fractional fixed point arithmetic.
- Contains continuous BIT, with fault indicator.
- Less than 10-second warm-up time.
- Indirect air, conductive cooling.
- Weight—40.0 pounds.
- Size—7.625 inches × 12.7 inches × 15.56 inches.
- Power required—115v, 3 phase, 400 Hz, 230 watts.
- 70 percent memory fill.

Two of the four multiplex buses interface with individual peripheral units, permitting selectable standby redundancy. Each multiplex bus set is capable of interfacing with up to 15 peripherals. There are twelve subsystems connected to the buses (six per pair), and five of these have their own computers. Some of the avionics equipment is hardwired. Bus numbers 1 and 3 are the primary buses; numbers 2 and 4 serve as their respective back-ups. The maximum data transfer rate is 45,000 words per second per channel (bus).

Two avionics devices, the BIT control panel and the avionic status panel, also interface with the CC on dedicated buses. Each of these panels is connected to the CC with its own transmission line. Because the signals transmitted are either one of two constant binary levels, transmission of clock is not necessary.

There is also a separate radar multiplex bus so that the aircraft can perform its mission in case of a loss of communication between the radar and CC. The present F-15 hardwired signal processor is being replaced by a programmable signal processor (PSP).

The Radar Warning Receiver (RWR) computer is also tied to the digital multiplex bus system. In addition to processing signals from the AN/ALR-56 RWR, the RWR computer controls the AN/ALQ-135 Internal Countermeasure Set (ICS) and the AN/ALQ-128 Electronic Warfare Warning Set. Together these components make up the F-15 Tactical Electronic Warfare System (TEWS).

3.0 F-16 AVIONICS ARCHITECTURE

3.1 Weapon System Overview

The F-16 "Fighting Falcon" is a lightweight, single seat, single engine multirole tac-

tical fighter configured for air-to-air and air-to-ground operation with primary emphasis on air superiority. It is an international fighter manufactured by General Dynamics Corp., with some final assembly and component co-production performed by a five-nation consortium. General Dynamics is responsible for the entire aircraft and support equipment. The U.S. Air Force total planned buy is 1388 aircraft with an additional 348 for the four European Participating Governments (EPGs)—Belgium, Denmark, Netherlands, and Norway. Current schedules indicate that the aircraft will be in production until the late 1980s.

The F-16 weapon stations permit carrying of up to 15,300 pounds of either conventional or nuclear ordnance. Its armament consists of one internal M61 20mm cannon and six AIM-9L Sidewinders, with potential to carry an additional four AIM-7F Sparrows. The F-16 is also planned to receive AMRAAM. The aircraft is scheduled to transition to Ogden ALC (OO-ALC) in October 1985. However, this date might slip. Since the F-16 advanced radar currently under development is the basis for the CMMR and the F-16 has the most advanced digital avionics suite of current Air Force tactical aircraft, more detail on its architecture, radar, and software is provided in Chapter Three than for the F-15.

3.2 Avionics Architecture and Hardware

A diagram of the F-16 distributed avionics system architecture is presented in Figure D-2. Digital communication among F-16 avionics subsystems is accomplished over a dually redundant multiplex bus system. Data is transmitted by a command/response control scheme at a 1MHz bit rate over half-duplex channels. Waveforms, timing, and word/message formats are as prescribed in MIL-STD-1553A. Data may be transmitted either between a bus controller and a remote terminal or between two remote terminals.

3.2.1 Multiplex System and Control

Primary multiplex control is performed by the Fire Control Computer (FCC) through a bus controller that operates on software commands stored in the computer's memory. The FCC initiates all information exchanges over the data buses by issuing command words to remote terminals to either transmit or receive data. It also determines whether Bus A or Bus B is to be used for the transmission. A backup bus controller function is resident in the Inertial Navigation System (INS). If the FCC is turned off or fails to pass its self-test/built-in-test, a discrete signal to the INS is turned off to indicate that the backup controller is to assume control. Under these conditions, the backup controller indicates all information exchanges over the bus and selects the bus to be used.

The remote terminals are integrated into the avionics subsystems that communicate over the multiplex system. Consequently, the individual entities and functions of the Multiplex Terminal Unit (MTU) and the Subsystem Interface Unit (SSIU) described in MIL-STD-1553A are absorbed into the design of each avionics unit. The remote terminal function in each subsystem is customized to be compatible with the subsystem in which it resides so that an optimum interface is attained.

A standby approach is used for management of the dual redundant data buses. Each of the twelve avionics subsystems is connected to both of the multiplex data buses. Subsystems monitor both buses for valid command words. A subsystem responds with data and/or status words only on the bus on which a command word is received. The bus con-

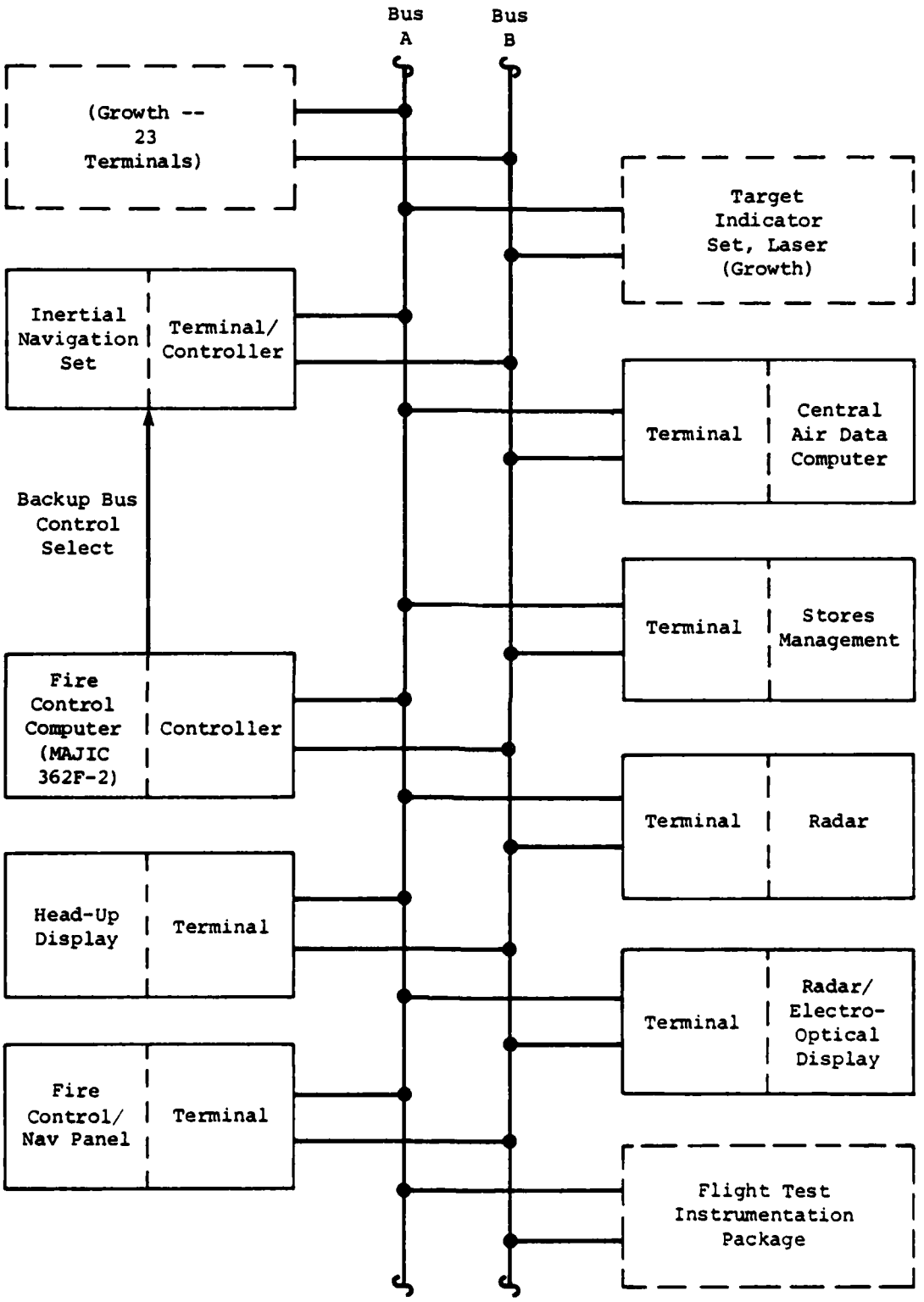


Figure D-2. F-16 AVIONICS ARCHITECTURE

troller may use either bus for any data transmission, but a given command is transmitted over only one of the two buses. If a subsystem is transmitting information on one bus and receives a valid command word on the other bus, the subsystem terminates its transmission on the first bus and responds to the command over the other bus. This feature permits the bus controller to deactivate a system element making uncommanded transmissions on a bus. The bus control function determines the bus to be used for each transaction by analyzing status words and checking for proper remote terminal responses. Plans are underway to add another dedicated multiplex bus (MIL-STD-1553B) for the F-16 display system.

Growth potential and flexibility are the principal attributes of the F-16's avionics architecture. Interface parameters can be added to or deleted from the data bus by changing the bus control algorithm and modifying the affected subsystem software.

The characteristics of the FCC and fire control radar computer (FCRC) are summarized in Table D-2.

Table D-2. F-16 FIRE CONTROL COMPUTER (FCC) AND FIRE CONTROL RADAR COMPUTER (FCRC), SOFTWARE AND HARDWARE CHARACTERISTICS

			FCC	FCRC		
				Initial	Update	Advanced
OFP	Size		26k	32k	36k	191k
	Language		J3B-2 (~80%) *Ass. (~20%)*	Assembly (100%)	Assembly (100%)	J-73 (~85%) *Ass. (~15%)*
	Developer		General Dynamics	Westinghouse	Westinghouse	Westinghouse
Host Computer	Memory Size	CORE	32k			
		EPROM		32k	48k	48k
		RAM		4k	16k	16k
		BORAM				256k
		ROM (CPU)	1024 40 Bit Words			
	Word Size		16 bits	16 bits	16 bits	16 bits
	Memory Fill	CORE	80%			
		EPROM		100%	~75%	~75%
		BORAM				~75%
	Manufacturer		DELCO (MAGIC 362F-2)	Westinghouse	Westinghouse (MILLI)	Westinghouse (MILLI-EP)

3.2.2 Fire Control Computer (FCC)

The FCC is a modular, general-purpose, stored program digital computer that is a modification of an existing design manufactured by Delco Electronics. The computer is used to implement computations for weapon delivery, energy management, serial digital bus control, navigation-related functions, and self-test.

The computer consists of a single line replaceable unit (LRU) containing 12 shop replaceable modules. It is composed of a solid-state microprogram-controlled, high-speed central processing unit, an input/output (I/O) section, and a core main memory. The microprogram is contained in 1024 40-bit words of read only memory (ROM). The detailed characteristics of the FCC are listed in Table D-3.

Table D-3. CHARACTERISTICS OF F-16 FIRE CONTROL COMPUTER

- 32,768, 17 bit (1 parity), words of memory storage, not expandable.
- Capable of interfacing with 2 channels and up to 33 peripherals.
- Capable of performing 250,000 operations per second.
- Data word length—16 bits.
- Requires no adjustments in aircraft.
- No memory loss during power transients.
- Full parallel operation.
- Uses fixed point and floating point two's complement.
- Contains continuous BIT, with fault indicator.
- Less than 10-second warm-up time.
- Indirect air, conductive cooling.
- Weight—27 pounds.
- Size—7.6 inches × 17.8 inches × 5.40 inches.
- Power required—115v, 3 phase, 400 Hz, 198 watts.
- 80 percent memory fill.

The I/O section includes a dual-redundant serial digital data bus interface and controller, discrete inputs, internal and external interrupts, and direct memory access. A support equipment connector makes it possible to monitor the internal operation of the computer on a real-time basis.

3.2.3 Relevant ECPs

We evaluated several software-intensive ECPs for the F-16 aircraft. The most applicable of these was ECP 0098, which is designed to update the software associated with the FCC's Operational Flight Program (OFP) and the Stores Mangement System (SMS) Operational Program (OP). The software changes contained in ECP 0098 are those that have been accumulated through avionics software-only changes as well as those occurring as a result of hardware modifications. Retrofit of these changes will update all aircraft through production number A116 and B51 to a common "Block II" configuration. Production incorporation of the change will occur for aircraft beginning with A117 and B52. The rationale for accumulating the changes in a block update was to provide for sharing to the maximum extent possible tasks common to all OFP and operating changes such as documentation, integration, retest and functional verification.

Two other ECPs involving hardware changes to the Head-Up Display Electronics Unit and the Fire Control Radar must be accomplished concurrently with ECP 0098 in order for the new software to function properly. The first of these, ECP 0206, incorporates the hardware changes necessary to allow AIM-9L missile seeker pointing symbology to be displayed on the Head-Up Display. The software changes required for this ECP have been incorporated in ECP 0098. The other ECP that must be accomplished concurrently is ECP 0116, which improves the capability of the Fire Control Radar. With this ECP, computer memory and speed increases are incorporated to provide the necessary ECCM and track capabilities against the threat. Changes include a 25 percent increase in both memory and

speed through the use of double density EPROMS and a 16k RAM. Increased operational capabilities are also being gained as a result of the above changes through the incorporation of additional radar modes (ACM Improvement, Boresight, Near Term Fire Control Update, etc.) that were identified during the DT&E testing utilizing the additional memory and speed. Software changes required by ECP 0116 are also being accomplished under ECP 0098. The scope of the changes in each system as a result of ECP 0098 is limited to a 2000-word task in the FCC and a 750-word task in the SMS.

4.0 F/A-18 AVIONICS ARCHITECTURE

4.1 Weapon System Overview

The F/A-18 "Hornet" is a single seat, twin turbofan, multi-mission fighter/attack aircraft manufactured by MCAIR. The total planned buy is 1366 production aircraft for Navy and Marine Corps use to replace the aging F-4 and A-7 fleet. The Navy's present F/A-18 inventory objective is highly contingent upon the eventual outcome of the Marine Corps AV-8B Advanced Harrier program currently in development. A newer aircraft than the F-15 and F-16, its Initial Operational Capability (IOC) is scheduled for March 1981.

The F/A-18 has nine weapons stations which permit carriage of up to 13,000 pounds of interchangeable ordnance. Its armament consists of one M61 20mm cannon, two AIM-9L Sidewinders and two AIM-7F Sparrows. The new AMRAAM is also slated for the aircraft.

The F/A-18 has the most advanced digital avionics architecture and the most advanced multi-mode radar (APG-65) of the three aircraft reviewed. The APG-65 is one of the CMMR candidate radars. Because the radar baseline has not been frozen, much of the radar software information provided in this section is generalized. These data were used to develop the CMMR baseline used in time quantitative analysis of the software sition strategies of Chapter Four.

4.2 Avionics Architecture and Hardware

The F/A-18 avionics suite shown in Figure D-3 has been designed using a fully distributed digital architecture. The aircraft's computational requirements are categorized as those which are "sensor oriented" and those of a "mission oriented" nature. The sensor oriented computations such as radar signal processing, inertial navigation, and air data are handled solely by processors in those subsystems. The required mission oriented computations are handled by two main mission standard avionics computers (the AYK-14 mentioned in Chapter 2). The characteristics of these computers are listed in Table D-4.

The all-digital nature of the F/A-18 makes it readily adaptable to changing threats and incorporation of new sensors and equipment. It represents the most highly integrated avionics arrangement of subsystems, considering any previous aircraft. It has more on-board computer memory than any other aircraft in the DoD inventory-including the E-3A, P-3C and B-1. Its method of integrating flight controls, armament, displays and controls and other subsystems, including automated communications control probably will be a dominant feature of future aircraft.

All subsystem interface with the two AYK-14 main mission computers is via four 1553A multiplex buses. The operation of these buses is the same as that described for the

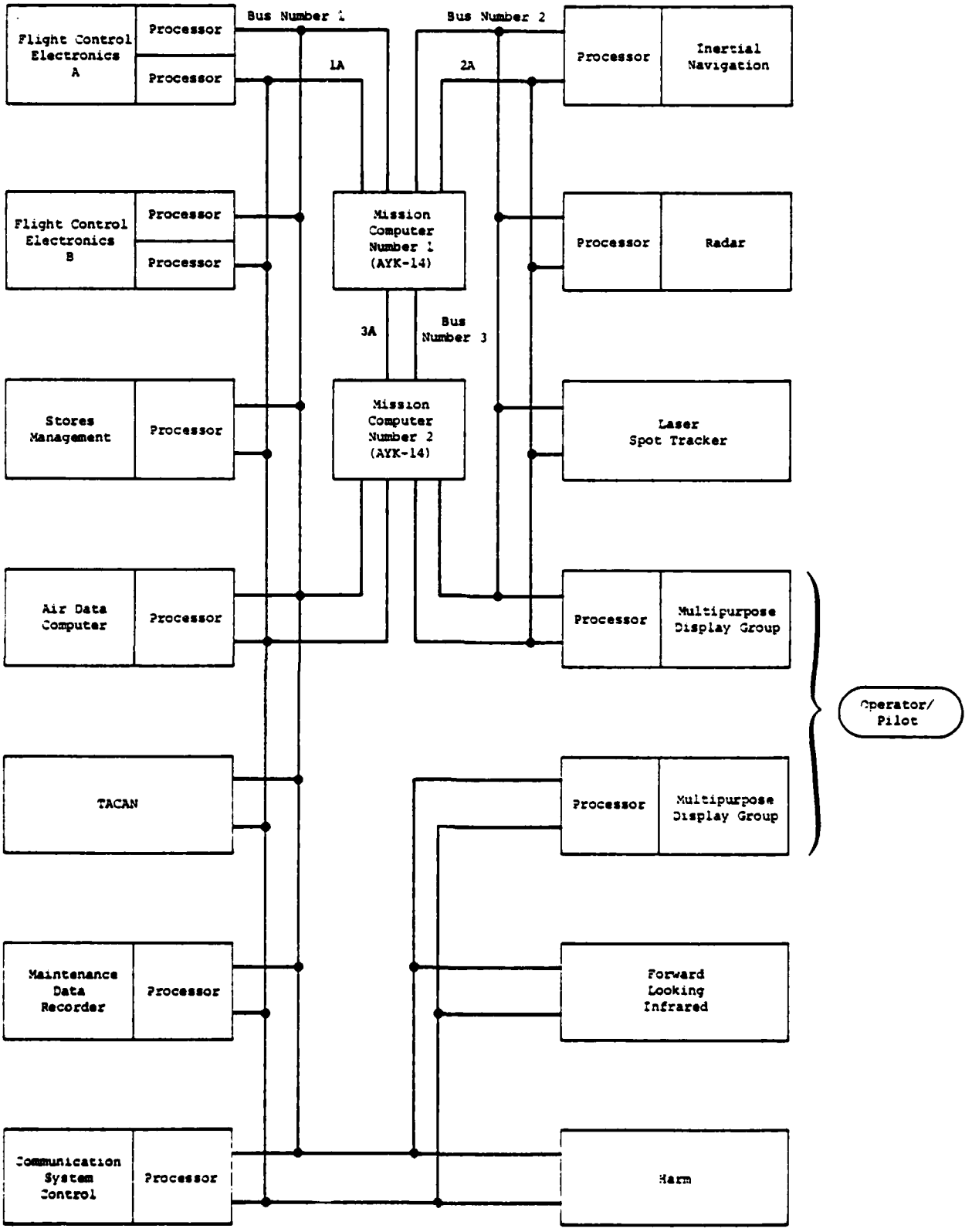


Figure D-3. F/A-18 AVIONICS ARCHITECTURE

Table D-4. CHARACTERISTICS OF AYK-14 STANDARD AVIONICS COMPUTER

- 65,536, 18 bit (2 parity), words of memory storage, not expandable due to physical size.
- Capable of interfacing with six channels and up to 32 peripherals.
- Capable of 400,000 operations per second.
- Data word length—32 bits.
- Requires no adjustments in aircraft.
- No memory loss during power transients.
- Full parallel operation.
- Uses fixed and floating point, two's complement.
- Contains continuous BIT, with fault indicator.
- Less than 10-second warm-up time.
- Air-conditioned cooling.
- Weight—40 pounds.
- Size—7.625 inches × 10.125 inches × 14.0 inches.
- Power required—115v, 3 phase, 400 Hz, 439 watts.
- 80 percent memory fill.

F-16. Two of the four buses are redundant. A third pair of channels (buses) performs the interface function between the main computers.

Two unique features of the F/A-18 avionics are a pair of digital flight control units with two processors each and a digital communications systems computer. Another unique feature is that the aircraft has been designed for both the fighter and attack roles since its inception. It can perform either of these missions without changing software. Of the thirteen subsystems tied to its 1553A buses, ten have their own processors.

The F/A-18 development and production contract has several unique design-to-cost and reliability and maintainability guarantees and incentives which provide for award fees rather than penalties. A production design-to-cost incentive is tied to a goal of 34 percent of the overall LCC. A 3.7 hour reliability for the aircraft Mean Flight Hour Before Failure (MFHBF) is guaranteed. The radar Mean Time Between Failure (MTBF) initially starts at 60 hours for the first production unit and grows to 100 hours by the 125th unit. Overall avionics MTBF is guaranteed to be 30 hours. There is a less than eleven hour Maintenance Man Hour Per Flight Hour (MMHPFH) guarantee for maintainability.

Because of these incentives and guarantees the prime and subcontractors have been given more liberal control of the weapon system, aircraft and subsystem design during prototyping and preproduction. Hardware and software specifications will not be frozen until the 21st production aircraft early in CY1982. A computer program product specification (CPPS) identifying avionics design and performance is maintained however.

APPENDIX E

REPRODUCTION OF TECHNICAL PERSPECTIVE # 31

Number 31

June 1980

A REVIEW OF COST-ESTIMATING PROCEDURES FOR DEFENSE SYSTEM SOFTWARE

Paul O. Chelson
William E. Thompson

1. INTRODUCTION

This technical perspective identifies and reviews methods used to estimate the life-cycle cost of defense system computer software. The need for valid cost-estimating procedures is critical because the investment and the risk involved in software development and support are enormous (Ref. 8, 38). No uniform software cost-reporting system is available to the DoD at the present time (30). Generally, software life-cycle costs are estimated and resources are allocated on an ad hoc basis.

This perspective provides an overview of some existing computer software life-cycle costing procedures, including a number of proprietary models developed by various private firms (2, 3, 5, 8, 9, 22, 31, 45, 48). It details what methods are available and what is known concerning effectiveness and past success. The annotated bibliography lists 61 references to papers, engineering reports, and Government documents related to computer software and computer system costing.

This perspective emphasizes cost estimating for embedded computer systems.* In these systems computer resources are incorporated into a larger system whose primary function is not data processing. Software used by embedded computer systems is concerned with all associated computer resources, a term used to describe the total hardware, computer programs, associated documentation, contractual services, personnel, and supplies required for computer development as well as for operations and support. Each of these resources is considered to be an element of the cost associated with computer system development and support.

---

*These systems are defined in Air Force Regulation 800-14 (13).

1.1 Overview of Software Costing

Software costs can be evaluated in two ways: before expenditure and after expenditure. Evaluation before expenditure requires prediction methods. Evaluation after expenditure requires methods for measuring and collecting cost information. Measurement of software costs has been limited to selected elements for a few programs. There has been no opportunity to use historical information for software costing, primarily because there is no effective and valid data base.

There are several different models for software costing, such as the rule of thumb -- 40 percent analysis and design, 20 percent coding, 40 percent integration and test -- accounting models, and stochastic probability models. Some models apply to one part of the software life cycle, while others apply to the entire life cycle.

The objectives of identifying procedures for estimating and reporting software-related costs are as follows:

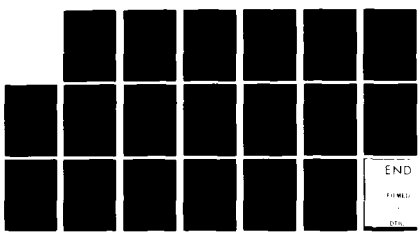
- To provide information for defense planning and budgeting
- To allow informed and critical evaluation of cost proposals
- To assist system project management in scheduling and controlling software funds
- To provide methods for detecting cost problems and potential budget overruns early in development
- To allow compilation of software cost data across systems to produce a historical data base
- To identify specific and measurable cost elements and relationships, including man-hours, computer time, calendar time, and documentation and training costs

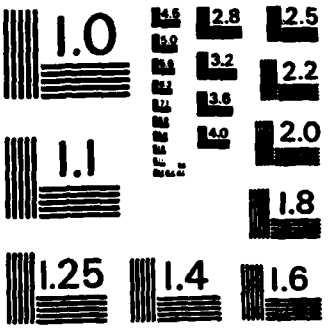
The current review of software costing addresses the entire life cycle. Development and implementation of changes to defense system software over the operational life of the system are now being recognized as a major cost element, perhaps exceeding the original software development cost in some cases. Because the design of software during development strongly influences the total life-cycle cost, the design concept must be able to facilitate efficient change (7, 32, 58, 61).

It is generally accepted that post-deployment software operational and support costs exceed development costs for major defense systems. Moreover, there are cost trade-offs among life-cycle phases. Where such partitioning of costs and trade-offs exists, there can be a discussion of the optimum allocation of resources to each phase. Such allocation may affect not only the total cost but the risk and schedule as well (54).

The planning and execution of a project largely determine which and how much of various resources are consumed. In a like manner, the planning

AD-A121 058 COMMON MODULAR MULTIMODE RADAR (CMMR) SOFTWARE 2/2
ACQUISITION STUDY(U) ARINC RESEARCH CORP ANNAPOLIS MD
S BAILY ET AL. MAR 81 2302-01-1-2291 F09603-80-G-3338
UNCLASSIFIED F/G 15/5 NL





and allocation of resources among software life-cycle phases affect the quality of the final product. Such relationships have been reported in Reference 34 and are developed in terms of quantitative measures in Reference 53. Nevertheless, the interrelationship associated with allocation of costs is difficult to quantify and validate. Although data from many similar software development projects are required, resources have been allocated and reported in different ways. Theoretical studies (34, 35) based on data analysis (5, 56) have established that future software life-cycle-cost studies must consider and report the allocation of resources.

1.2 DoD Policy and Regulations

MIL-STD-490, *Specification Practices* (18), provides official guidelines for both hardware and software components. It is often supplemented with more detailed directives for computer programs, program documentation, and operator and maintenance manuals. AFR-800-14, *Volume II, Acquisition and Support Procedures for Computer Resources in Systems* (13), provides official guidelines and required procedures that shape the entire software life cycle and that should be considered by cost analysts for Air Force software projects. MIL-STD-483, *Configuration Management Practices for Systems, Equipment Munitions, and Computer Programs* (17), provides a supplement to MIL-STD-490. It defines the content and organization of software specifications and development procedures. It also defines the content and procedures for Engineering Change Proposals for computer software (Appendix XIV), together with software change status control tools (Appendix VIII). MIL-STD-1521, *Technical Reviews and Audits for Systems, Equipment and Computer Programs* (19), prescribes requirements for technical reviews and audits in conjunction with software specifications and documents defined in MIL-STD-483.

When these and other similar standards and directives are adhered to in software acquisition projects, the costing task is more effective and accurate. These directives largely determine the relative allocation of resources between software acquisition phases. They also allow more systematic and consistent comparisons of software development efforts.

2. SOFTWARE LIFE-CYCLE RESOURCES AND COST FACTORS

The life cycle of defense system software consists of many phases and activities: conception, validation, analysis, design, coding, checkout, test and integration, transition, and, finally, operation and support. These elements can be grouped and combined in numerous ways (23, 31, 33, 55, 56). Because the focus of this perspective is software cost, the life cycle has been divided into three major phases: specification and design; production, integration, and test; and, finally, operation and support. These are depicted in Figure 1.

Because the three phases have different goals and objectives, the software activities, tasks, functions, and required resources are

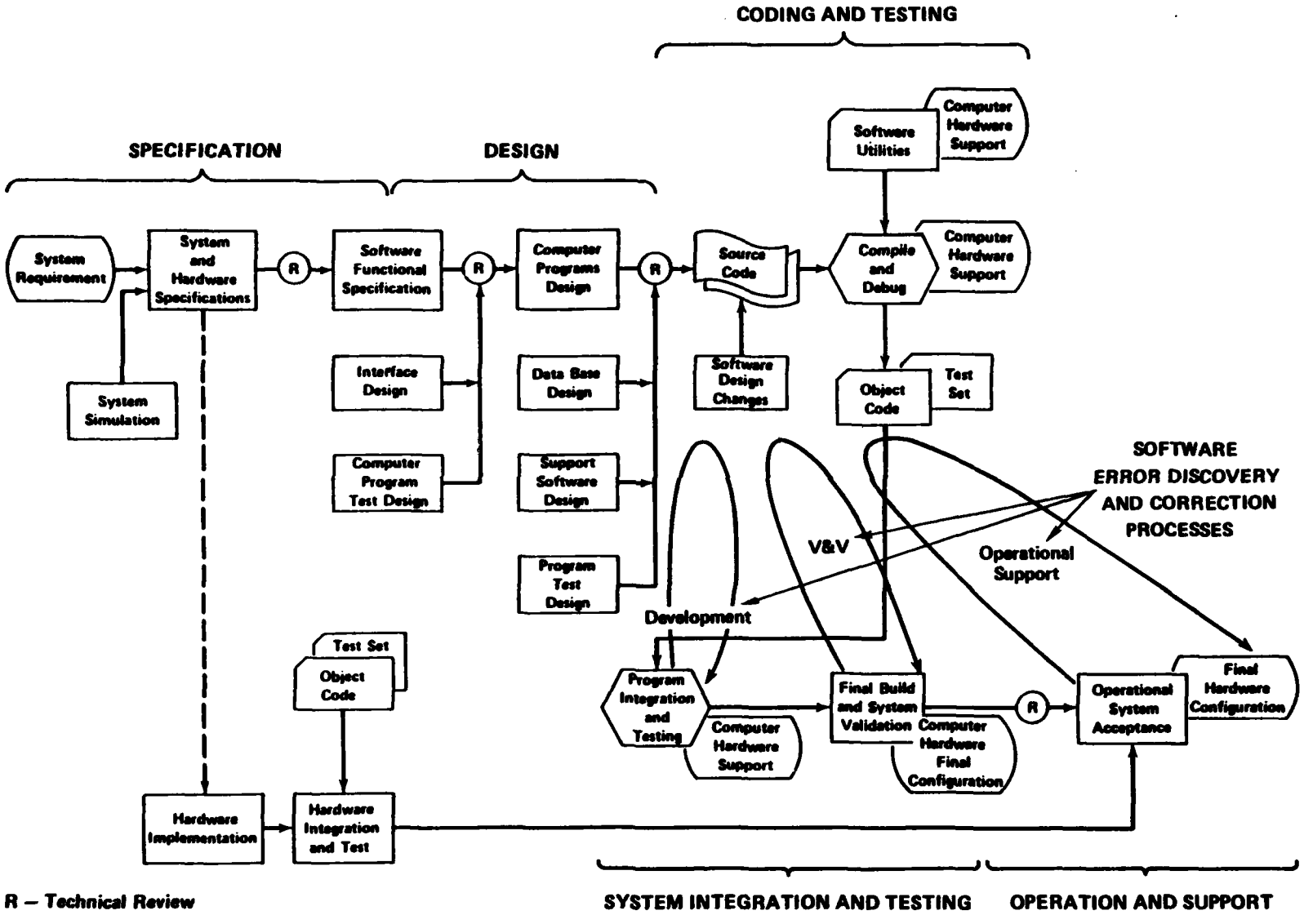


Figure 1. COMPUTER SOFTWARE DEVELOPMENT PROCESS

different. This difference in required resources results in different cost relations; therefore, each phase must be addressed and analyzed separately when the cost is being considered. To determine a method for establishing costs, the required resources must be understood. The following subsections briefly describe the purpose and required resources of each phase.

2.1 Specification and Design Phase

The first phase of the development process is the completion of a comprehensive system specification and the definition of the software mechanization required. This process starts with a top-down software system engineering effort to identify guidelines, conventions, constraints, interfaces, inputs-outputs, memory, training, and other technical aspects of the system. It establishes management requirements and plans for configuration management, quality assurance, verification/validation, testing, and acceptance.

After the specification is completed, the design effort further defines the system by means of the following:

- Identifying computer hardware and software functional requirements
- Defining interface data and control transfers
- Specifying hardware, functional, and data constraints
- Defining detailed performance and test specifications
- Defining software constraints
- Specifying all deliverables
- Establishing management requirements

The resources for this phase are dominantly labor, especially senior personnel. This major effort involves paper analysis, planning, top-down design, architectural layout, and definition of the support philosophy. The actual amount of coding developed is minimal.

In defense applications, the specification and design phase results in computer program configuration item (CPCI) specifications (17). These specifications consist of two parts. Part I specifications are performance and design requirements. Part II specifications define product configuration and include detailed technical descriptions of the CPCI. The software design phase begins with the baselining of Part I specification and ends with the development of the Part II specification for the software.

A Preliminary Design Review generally is conducted when the Part II specification is complete. At this point, the software design team identifies distinct CPCIs. The Part II specification provides "as-built" documentation. At the end of the Critical Design Review the CPCIs are broken down into computer program modules, the level at which code production begins.

2.2 Coding, Integration, and Test Phase

The second phase consists of the coding subphase and the integration and test subphase. The first subphase is the production of the software end items that must be integrated into the system. All the planning and support functions, such as configuration management, detailed procedures for testing, application module code, detailed software architecture, facilities and resources for testing, support establishment, and a working management plan, must materialize in this subphase. This subphase requires the largest labor force and numerous computer resources, including dedicated and general-purpose computers. Low-level testing and debugging are the first signs of progress. As this subphase progresses, the schedule becomes a more dominating factor and the trade-off of time may have a very nonlinear relationship to cost.

In the integration and test subphase, such system support functions as quality assurance, configuration management, validation/verification, and acceptance come to the forefront. The cost relationships become even more nonlinear than in the coding subphase. Error correction and redesign become more expensive, and the available resources are confined so that the probability of schedule slippage rises. The possibility of overtime increases because of the probability of schedule slippage. Transition is initiated, requiring involvement by Government users and supporters. Documentation deliverables often are behind schedule and commonly are incomplete. All these factors draw on confined resources, result in higher cost ratios versus calendar time, and require overtime and the attention of senior personnel.

2.3 Operation and Support Phase

The operation and support (O&S) phase of the software development process is commonly the longest phase. The monthly recurring cost generally is lower, but the O&S cost may exceed the development cost over the duration of the operational cycle (1, 2).

Many costs associated with O&S, such as support investment, training, documentation, support facility establishment, and management and logistic plans, may be budgeted for and associated tasks completed in the two previous phases. The extent to which this is true will have a major influence on the cost incurred in the O&S phase.

After fielding the system, the user commonly requires modifications to the system as a result of poor performance, reliability, maintainability, availability, or changed requirements. These modifications result in Engineering Change Proposals (ECPs). These changes can be more costly than the original system.

3. SOFTWARE COST ESTIMATING FOR SPECIFICATION AND DESIGN

Software specification and design costing involves the estimation of manpower, facilities, and calendar time to complete the Part II

specification during the Critical Design Review. Generally, a homogeneous mix of labor types can be assumed for the entire specification and design process.* For this perspective, we reviewed several methods used in cost estimating software specification and design.

3.1 Electronics Systems Division Method

The costing procedure used by the Electronic Systems Division (ESD) of the Air Force identifies five different types of software (14, 53):

- Operational flight software
- Electronic warfare software
- Command, control, communications, and intelligence (C^3I) software
- Simulation and trainer software
- Automatic test equipment software

This method assumes that certain measures of total complexity and logical difficulty can be identified with each of these software types. Equations are provided for both the specification and design activities for each of the five types of software.

For the specification activity, the size of the particular software development of a given type is specified by a value, X_1. The size, X_1, is associated with estimates of the total number of instructions (in source or object code) in the "final build," the final application software package. A cost-estimating relationship, specification man-hours = $a_1 X_1^{a_2}$, is assumed. The constants, a_1 and a_2, are derived by regression fit to many previous efforts of the given type of software.

For the design activity, new constants, b_1 and b_2, are identified for each of the five types of software, together with a new measure of size, X_1, which is defined as the total number of instructions in the final build less the number of instructions taken from existing code. The manpower estimates for the software design activity is design man-hours = $b_1 X_2^{b_2}$.

3.2 Software Life-Cycle Management (SLIM) Method

More complex but similar relationships have been derived, including allowed development time, state-of-the-art technology, and risk factors. These relationships provide a means to trade off resource requirements against schedules and risk. The SLIM cost-estimating package was developed by Quantitative Software Management, Inc., of McLean, Virginia (45). In this method, the size, X, is estimated from development time, t_d,

---

*This assumption is consistent with the level of approximation achieved elsewhere in the costing process.

state-of-the-art technology, C_K, and a difficulty gradient, $||\nabla D||$, by using the formula

$$X_i = C_K \, ||\nabla D||^{1/3} \, t_d^{7/3}$$

where C_K and ∇D are obtained by empirical means as shown in Reference 45.

3.3 PRICE-S Method

The Price Software (PRICE-S) model developed by RCA is one of the most complex and widely used life-cycle costing models available today (48). It uses parametric modeling to forecast both software costs and schedules. It is designed to cover all types of software development, including business systems; communications, command, and control systems; and avionics and space systems. This wide range of application is achieved, according to RCA, by techniques that are "proprietary but are substantial improvements over customary cost-estimating relationships." In addition to standard cost and schedule estimates for each phase of software development, PRICE-S provides sensitivity and schedule-effect analyses, monthly cost and progress summaries to support project management, and budgeting and risk assessment.

The specification and design part of PRICE-S uses input parameters related to typical program sizes (number of source statements), required schedules, project familiarities, design phase requirements, amount of existing design, and hierarchical design/functional flow structure. PRICE-S assumes that every software project can be completely characterized by a set of about 50 parameters. Even if this is true, the user often has difficulty providing the required values for these measures before a project begins. The estimation of resources required for the specification and design phase is especially difficult.

4. COSTING FOR SOFTWARE CODING, INTEGRATION, AND TEST

Coding begins after the Critical Design Review and does not end until all elements of the final build are in the integration and test phase. The function of coding is to write and debug the computer programs required by the Part II specification.

Users of nearly all of the available coding, integration, and test costing models encounter one or more of the following problems:

- Accurate estimation of the number of code statements is difficult.
- Programmer productivity depends on many factors that cannot be prejudged.
- Accurate prior estimates of the impact of design changes cannot be obtained.

- The adequacy of the hardware cannot be accurately determined prior to code production, in many cases resulting in critical and time-consuming coding procedures and possible rejection of major coding results.

4.1 Air Force Systems Command Method

The equation used by the Air Force Systems Command (33) is coding and debugging man-hours = $c_1 X_3^{c_2}$, where c_1 and c_2 are constants determined from previous experience with similar types of software and X_3 is the number of source statements. The dependence on type of software is less critical than in the specification and design phase. Nevertheless, the dependence on experience levels and coding techniques may be critical.

4.2 System Development Corporation Model

The System Development Corporation model (29) was derived from 169 data points by multiple regression. It uses the following parametric equation for estimating coding costs:

Coding man-months = K × PM

where K is a constant of proportionality determined by the units used and PM is a linear function of the following empirical factors, each of which is a constant between 0 and 1:

f_1 = Lack-of-requirements factor

f_2 = Stability-of-design factor

f_3 = Percentage storage/retrieval instructions

f_4 = Number of subprograms

f_5 = Programming language factor

f_6 = Business application factor

f_7 = First-program-on-computer factor

f_8 = Concurrent hardware development factor

f_9 = Random-access-device-used factor

4.3 IBM Method

The IBM model (42, 58) has been reported to achieve accuracies as follows (43):

- 14 percent accuracy in man-months
- 22 percent accuracy in development dollars

The following quantitative information has been reported (43) from IBM's experience with costing models and their results:

- A 9 percent annual increase in programmer productivity has been observed over five years.
- A 30 percent average difference in productivity has been observed between programmers familiar and those not familiar with a project.
- A 30-to-1 variation in productivity has been observed between individual programmers with similar experience and background.
- A fourfold increase in computer time is required to support software development for new and unfamiliar hardware.

IBM uses a software costing strategy of initially overestimating 15 percent on personnel and 40 percent on overall cost. As the project progresses, these overestimates are narrowed as appropriate (43).

The IBM model assumes that software integration and test begins before coding is completed. As computer program configuration items (CPCIs) are completed, they are checked out and combined into the final build. Generally, the integration and test phase requires using the computer and often includes critical components of system hardware other than the computer. As in design and coding, the general equation, integration and test man-hours per CPCI = $d_1 X_i^{d2}$, is used in the IBM model. Here X_i is the size (in source statements) of the i^{th} CPCI in the final build, so that for a final build consisting of N CPCI's:

$$\text{Total integration and test man-hours} = \sum_{i=1}^{N} d_1 X_i^{d_2}$$

The assumption that the constants d_1 and d_2 are independent of the module size, X_i, should be verified in each application.

In addition to the man-hours allocated above, a formal software evaluation demonstration or acceptance test may be imposed. In this case the labor mix may not be homogeneous; it may consist of software engineers and quality assurance personnel as well as programmers. The total resources for this effort are best estimated on an ad hoc basis. Labor cost, total computer time (both test and posttest analysis), and review and reporting costs may approach 25 to 35 percent of the total software development cost for large, complex systems. The number of errors in the test build and possible system redesign requirements are often the driving factors in final costs. These factors generally cannot be expressed in generic cost-estimating relationships as simple as that given above.

4.4 General Research Corporation (GRC) Model

The GRC cost-estimating method (33) suggests using an aggregated model for integration and test of a final build of types j=0, 1, or 2 software of size Z_1, which can be expected to require Z_2 changes. For example,

integration and test for entire build in man-months = $h_j Z_1 (1+h_j Z_2)$ where h_0, h_1, and h_2 are constants related to the three software types identified, Z_1 is a measure of size (number of instructions) of the build or portion of interest, and Z_2 is the number of formal engineering changes required. The difficulty in using this model is that little valid information is available for evaluating h_0, h_1, and h_2 for the three types of software identified.

5. SOFTWARE COST-ESTIMATING FOR THE O&S PHASE

Several sources were reviewed to gain insight into existing software O&S cost-estimating techniques. The types of models and algorithms used are presented in this section. As of this writing, there is no viable model for the defense community. Each model has its deficiencies.

5.1 Putnam/QSM Model (45, 46, 47)

The Putnam/QSM, model was said to be verified against 150 projects. It takes a macro-statistical estimating approach by using only 3 to 5 factors. The lines of code, time schedule, and level of complexity are major factors.

5.2 Percentage of Development Cost

Development costs are often assumed to account for some fixed percentage of the software life-cycle cost. Examples of this approach where O&S costs are spread across the O&S period are given in References 26, 46, 48.

The percentage of software life-cycle costs that represents O&S may range from 40 percent to 400 percent of the development costs, depending on the researcher and data base used. These percentages are derived from previous experience on similar software programs. There is too wide a variance and little if any direction as to site, complexity, or scope to depend on any of these models for accurate estimates.

5.3 Single-Equation Models

The following single-equation models were taken from Reference 26. These models appear to be tailored to specific programs and do not directly address parameters such as management and facilities.

- General Research Corporation Model

  $$\text{Cost} = 1{,}740 \times \text{AIMH} \times [(-0.00111 \times \text{KK}) + 0.01611]$$

where

Cost = Cost in dollars of correcting software in year KK after delivery

AIMH = Number of instructions

KK = O&S year considered, counting number of years after delivery

- Boeing Constant Level of Maintenance Model

  Man-years = I/10,000 (for E-3A)

  Man-years = I/1,685 (for F-16)

where

I = number of instructions

Because the O&S phase is multivariant and incorporates many features not related to lines of code, no single-equation model is likely to succeed. Furthermore, no external validation of these models has been found in the literature.

5.4 Accounting Models

The following examples of single-equation models are from Reference 26. The first model addresses the O&S part of the life cycle. The second model was used during development of the B-1 bomber to cost-estimate O&S.

In the Boeing Computer Resources Integrated Support Plan (CRISP) model

$$LCC = S_1 + S_2 + S_3 + S_4 + S_5 + S_6 + S_7,$$

where

LCC = Life-cycle costs, assuming a 10-year cycle

S_1 to S_7 = Elemental costs

Because S_4 is software support equipment maintenance cost and S_5 is software support cost, they are explained in detail:

$$S_4 = \frac{(NC)(ANIC)(SPI)(MMS)}{MHM}$$

where

NC = Number of changes

ANIC = Average number of instructions per change

SPI = Shifts per instruction

MMS = Maintenance man-hours per shift

MHM = Man-hours per month conversion factor

$$S_5 = (LC)(MMC)$$

where

LC = Lines of code

MMC = Man-months per line of code

In the Rockwell CRISP Model

$$SW = PR + VER + CM$$

where

SW = Software man-months for the operational and support phase of a program

PR = Programming man-months

VER = Verification and validation phase man-months

CM = Configuration management man-months

and

$$PR = NIF \times IPC \times \frac{[NR \times MHIN + (1-NR) \times MHIR]}{MHM}$$

where

NIF = B-1 lot variation corrective factor (based on block changes per year, common routines, and lot-buy quantity)

IPC = Instructions per block change

NR = Ratio of new instructions

MHIN = Man-hours to program new instructions

MHIR = Man-hours to program revised instructions

MHM = Man-hours per month conversion factor,

and

$$VER = NIF \times IPC \times \frac{[PI \times MHIN + (1 + TSF) \times TMMI \times LSF]}{MHM}$$

where

PI = Changes per instruction

TSF = Test support factor

TMMI = Test man-hours per instructions

LSF = Laboratory support factor,

and

$$CM = NIF \times IPC \times \frac{[NR \times MCM + (1-NR) \times MCR]}{MHM}$$

where

MCN = Man-hours per new instruction

MCR = Man-hours per revised instruction

Accounting models often rely on data that are not available early enough in the program to be employed for cost estimates.

5.5 Engineering/Accounting Model

An engineering/accounting model is a deterministic approach in which the cost elements are identified from a bottom-up approach and are combined in the appropriate mathematical manner. The following equation is an example:

$$\text{Year software O\&S cost} = \{[(\text{Failures per Year}) \times (\text{Man-hours per Failure}) + (\text{Design Changes per Year}) \times (\text{Man-Hours per Design Change})] \times (\text{Average Man-Hour Labor Rate}) + (\text{Total Cost of Support Facilities})/(\text{System Life, Years})\} + \{[(\text{Failures per Year}) + (\text{Design Changes})] \times (\text{Average Support Computer Hours per Action}) \times (\text{Cost of Computer Hours}) + (\text{Total Yearly Management Cost})\}$$

Intermediate variables can be developed by using regression analysis techniques, for example:

$$Y = b_1X_1 + b_2X_2 + \ldots + b_nX_n + c$$

where

Y = Intermediate variable (for example, computer hours, cost of computer hours, man-hours)

X_i = Independent variable (for example, program size, complexity, programming language)

b_i and c = Relating coefficients determined by the regression analysis

(Note: Nonlinear functions may also be used, i.e., logarithmic for skewed or nonlinear data distributions.)

Several variations can easily incorporate the effects of changing military objectives, warranty, and guarantees. Regression analysis is strongly dependent, however, on a data base, which to date has not materialized. As a result, this type of model commonly is influenced by ad hoc speculation on the cost of several variables.

5.6 Proprietary Models

The government has leased several proprietary models. Two of these are the Quantitative Software Management, Inc., Software Life-Cycle Management (SLIM) Method (43, 44, 45) and the RCA PRICE-S Model (46). Both these models were originally designed to estimate the cost of the software development phase. Nevertheless, they have or will have some capability to estimate the O&S costs of software. The SLIM Model is more an accounting model than a predictive model, and the relationship between development costs and O&S costs is fixed. Little is known about the RCA PRICE-S3, an extension of the RCA PRICE-S Model, but it is thought that it has some O&S cost-estimating capability.

5.7 Comment on Software O&S Cost-Estimating Techniques

The techniques reviewed are varied in structure and in types of input data utilized, and in some cases are tailored to specific types of systems.

Comments in Reference 26 include:

> "There are few useful tools available to engineers and cost analysts to help them predict the operational and support costs associated with computer software accurately (i.e., within about 25 percent of actual costs)."

The experience of SDC in an effort to improve life-cycle cost analysis by developing software cost-estimating relationships (reported in Reference 27) is typical. They surveyed Air Force Logistic Centers (AFLCs) to examine cost and technical data associated with the maintenance and modification of the functional areas of embedded software for which an AFLC has maintenance responsibility. The results indicate that no cost-collection mechanism is currently available for any of the maintenance and modification areas and that the cost drivers (i.e., technical requirements that affect schedule and performance) are different for each category of embedded software.

6. CONCLUSIONS

6.1 Summary of Status of Software Costing

This perspective has presented an overview of various approaches and techniques that have been used or are being used. There is no single dependable and precise method for determining how long a software project will take or how much it will cost. Although we cannot identify or recommend the "best" method, we can recommend that the user follow certain basic guidelines in attempting to obtain good estimates of software costs and schedules. These guidelines are presented in Table 1.

6.2 Future Development of Software Costing

After many years of experience, the engineering and development community has not completely solved the problem of estimating costs for

Table 1. ARINC RESEARCH GUIDELINES FOR SOFTWARE COSTING

1. Learn as much as possible about the project and its goal prior to costing. In particular, determine if possible:

   - Similar projects performed in the past, their cost, and the applicability of the lesson learned to the present project
   - Special constraints related to software specification, coding production, testing, and operational support
   - Trade-offs that are available in implementing hardware-software functions and extent to which these trade-offs will be maintained as viable options as the project develops
   - Uncertainties remaining in system requirements, project implementation, and final product application
   - The extent to which project management and contract support are motivated to produce on schedule and within budget; also, their previous performance if contractors have been identified
   - Contingencies in the event of severe schedule slippage, poor product performance, or budget overruns

2. Complete at least two software cost estimates using different methods. Examine differences in the estimates in detail and determine why they are different.

3. Submit at least two final cost estimates -- high and low -- and document all assumptions related to the reported cost elements in each case.

4. Recognize that software costing results in the past have been subject to 20 to 60 percent error in the best of cases, and frequently have been in error by as much as 200 to 400 percent. Generally, the error has been in cost estimates that were low. Most of the costs associated with software development are people costs, and the tendency is always to plan initially for utilization of all resources in the original concept, leaving unexpected contingencies to demand additional commitment. (Experience suggests that the initial commitment of resources should not exceed 75 percent of total resources.)

hardware and system development. Software development is a new and fast-changing technology. New methods and techniques for estimating software costs will continue to be presented, and existing methods will be changed and improved; but there is no reason to believe that a major breakthrough will deliver greatly improved costing methods in the next five years. Better software cost estimation and cost control are more likely to result from improved software specification, code production, and operational maintenance techniques than from improved costing formulas and procedures.

REFERENCES AND BIBLIOGRAPHY

1. D. S. Alberts, "The Economics of Software Quality Assurance," *AFIPS Conference Proceeding,* Volume 45, 1976, 433-442.

   This paper relates quality assurance in development to maintenance costs of software. It states that 50 to 60 percent of software life-cycle costs are maintenance-related.

2. ARINC Research Corporation, *Navy O&S Cost Estimating Techniques for Software,* NAVSTAR User Equipment Support, Contract No. F04701-78-C-0124, June 1979.

   This report analyzes and reviews O&S costs for software systems. It presents quantitative software model data from the GPS Baseline cost estimate. It recommends a particular software cost-estimating procedure for Navy O&S. The following software O&S cost models are presented: percentage of development cost, single CER equation model, accounting single equation model, engineering/accounting model, ESD methodology (Doty Associates model), and RCA PRICE-S model.

3. J. D. Aron, "Estimating Resources for Large Programming Systems," FSC-69-5013, Federal Systems Center, IBM, 1969, 1-21.

   This report presents the method used for IBM at Federal Systems.

4. V. R. Basili and M. V. Zelkowitz, "Operation of the Software Engineering Laboratory," *Second Software Life Cycle Management Workshop,* 21-22 August 1978, 162-166.

   This paper emphasizes models for resource estimation, the analysis of error and change data, and program complexity measures. Data were collected to evaluate software development disciplines, models, and measures. Cumulative costs for large-scale software development have been shown to approximate the curve $K\ (1-e^{-at^2})$, where K is the total project cost and t is the elapsed time into development.

5. L. A. Belady and M. M. Lehman, "A Model of Large Program Development," *IBM Systems Journal*, Volume 15, No. 3, 1976, 225-252.

Two models are developed to assist understanding the nature of software maintenance. The first model describes the internal distribution and propagation of errors in a computer program, which are eliminated by a specialized programming team. The second model expands the role of the team to make decisions as to what actions should be performed.

Model 1 provides a measure of complexity as a result of aging. Model 2 relates costs to the total activities of error elimination and program complexity. An important conclusion is that program complexity will increase continuously because of maintenance activity. Thus, the primary consideration in maintenance should be to develop better techniques to reduce unexpected side effects that tend to increase complexity.

6. D. E. Bell and J. E. Sullivan, *Further Investigations Into the Complexity of Software*, Mitre Corporation Technical Report 2874, Volume II, 30 June 1974.

This report presents a useful background description of complexity factors.

7. R. K. E. Black, "Effects of Modern Programming Practices on Software Development Costs," *Proceedings Fall COMPCON 77*, September 1977, 250-253.

This report presents basic findings but no general quantitative methods. It stresses that programming is a "people-oriented" activity.

8. B. W. Boehm and R. W. Wolverton, "Software Cost Modeling: Some Lessons Learned," *Second Software Life Cycle Management Workshop*, 21-22 August 1978, 129-132.

This paper summarizes lessons learned in developing a software cost-estimation model for TRW. The objective was to provide an aid in developing cost estimates for competitive proposals on large Government software projects. The paper presents: a suggested work breakdown structure to define which costs are included, estimator fidelity versus project size, examples of TRW cost-model performance, a list of factors used in the cost model, effect-of-complexity factor, an example of a complexity scale, and TRW cost-factor ratings.

9. G. E. P. Box and L. Pallesen, *Software Budgeting Model*, Mathematics Research Center, University of Wisconsin, Madison, February 1977.

This model uses the adaptive filtering technique developed by G. E. P. Box to fit actual overall code production rate as a function of time into development.

10. F. P. Brooks, *The Mythical Man Month*, Addison-Wesley Publishing Co., Reading, Massachussetts, 1975.

This much-quoted book describes software development and costing problems. It does not give detailed or specific costing techniques, but it provides a readable introduction to the problems of developing and costing software. Brook's Law -- "adding manpower to a late software project makes it later" -- is an example of one of the many quotable axioms.

11. J. R. Brown, *Practical Strategies for Developing Large Software Systems*, Addison-Wesley, Reading, Massachussetts, 1975.

This book presents methods of producing better software at lower costs. It describes cost considerations in a qualitative way.

12. Department of the Air Force, Air Force Regulation 300-2, *Management of the USAF Automatic Data Processing Program*, 19 August 1977.

This document consolidates procedures that apply to acquisition of automatic data processing systems.

13. Department of the Air Force, Air Force Regulation 800-14, *Acquisition and Support Procedures for Computer Resources in Systems*, Volume II, 26 September 1975.

Volume II consolidates procedures that apply to acquisition and support of computer resources in systems. It defines terms, establishes a software life-cycle concept, and identifies planning and management procedures.

14. Department of the Air Force, Electronic Systems Division, *Software Acquisition Management Guidebook Series*, Defense Documentation Center/ NTIS accession, 1976.

This 17-volume series provides an interpretation of military regulations and standards that apply to the acquisition of computer resources for Air Force electronic systems. Volume 14, *Software Cost Estimates and Measurement*, AD/AO 55574, presents a description of existing software cost-estimating techniques.

The Air Force Aeronautical Systems Division (ASD) has produced a similar series. The first guidebook deals with acquisition of computer resources in avionics systems and the second with acquiring computer resources for ground support systems.

15. Department of the Air Force, Electronic Systems Division, *Software Acquisition Management Guidebook Series: A Review of Software Cost Estimation Methods*, ESD-TR-76-271, August 1976.

This useful overview of methods includes a list of references.

16. Department of the Air Force, Electronic Systems Division, *Software Acquisition Management Guidebook: Software Development and Maintenance Facilities,* ESD-TR-77-130, April 1977.

This report is useful in identifying facility-related cost elements of software development and support.

17. Department of Defense, MIL-STD-483, *Configuration Management Practices for Systems, Equipment, Munitions, and Computer Programs,* 31 December 1970.

Appendix IXVI, "Computer Program Configuration Item Specification," provides instructions for preparation of detailed specifications for each computer program configuration item (CPCI). Preparation must be in accordance with MIL-STD-410. These instructions are applicable to each phase of development and cover computer program development from conception to operational support.

18. Department of Defense, MIL-STD-490, *Specification Practices,* 30 October 1968.

19. Department of Defense, MIL-STD-1521, *Technical Reviews and Audits for Systems, Equipments, and Computer Programs,* 1 September 1972.

20. Department of Defense, MIL-STD-1679, *Weapon System Software Development,* 1 December 1978.

This standard contains requirements that are applicable in Government contracts for the design and development of weapon system software. A standard specifically addressing weapon system software is necessary because of factors that are not common to general software or that carry a significantly different degree of emphasis. Major factors are criticality of performance, changing operational requirement, and life-cycle cost.

21. D. L. Doty, P. S. Nelson, and K. R. Stewart, *Software Cost Estimation Study: Guidelines for Improved Software Cost Estimating,* Volume 2, RADC-TR-77-2206, Griffiss Air Force Base, N.Y., August 1977.

This final report identifies more than 100 factors related to the cost of software development. Of these, 42 were found to be significant, but only about 29 of these were judged to be quantifiable. The study team reported that most of the errors in estimating costs result from underestimates of size, sometimes by a factor of three or more. It recommends that Air Force analysts be aware that size-estimate errors may be more than 200 percent in the conceptual phase of a project, 100 percent at the Request for Proposal stage, 75 percent at preliminary design review, and 50 percent throughout the remainder of development.

22. L. W. Dowdy and A. K. Agrawala, "Parametric Instabilities in Computer System Performance Production," *AFIPS Conference Proceedings*, Volume 48, 1979, 51-55.

Continual computer system upgrading is not generally desirable, and improvement efforts can lead to costly problems in software support. This paper provides data inputs for cost-estimating the maintenance of computer systems and software.

23. EG&G Washington Analytical Services Center, *Interim Guide for Preparation of Cost Estimates for Tactical Software Programs*, Rockville, Maryland, March 1976.

This document provides guidance in structuring and developing a cost estimate for tactical software programs. Instructions are provided for addressing associated hardware costs, coding costs, and related support and services. A description of the software development process and a work breakdown structure for development and maintenance of tactical software progress are given for a particular case.

24. I. R. Elliott, "Life Cycle Planning for a Large Mix of Commercial Systems," *Software Life Cycle Management Workshop*, 21-23 August 1977, 203-216.

This paper describes the resources required to support a large number of commercial software systems. The total amount of source code considered was 1,198 source lines, a very large system. The observed production rate, was 3,000 source lines of new code per man-year. Every 20,000 lines of code require one man for support. The number of system support personnel was 25 percent of the staff required in the first year of development.

25. A. Endres, "Analysis and Causes of Errors in System Programs," *Proceedings of the International Conference on Reliable Software*, April 1975, 327-336.

This paper describes the frequency and types of errors found in system programs. It presents useful data for estimating error-correction costs in post-deployment. The paper suggests that maintenance achieves a steady-state value and remains there for the entire life cycle.

26. D. V. Ferens and R. L. Harris, *Avionics Computer Software Operations and Support Cost Estimation*, Air Force Avionics Laboratory, Proceedings of the IEEE National Aerospace and Electronics Conference, NAECON, Volume 1, May 15-17, 1979, 296-300.

This paper presents an overview of O&S cost methods and the initiatives of the Air Force Avionics Laboratory in this area.

27. M. Finfer, "SDC Software Cost Experience," *Proceedings, NSIA Software Conference*, Buena Park, California, 13-15 February 1979, 315-329.

This paper reports on O&S costs in connection with an ongoing software cost-estimating study for Air Force Logistic Centers (AFLCs). To improve life-cycle-cost analysis, SDC surveyed AFLCs to examine cost and technical data associated with maintenance and modification of five functional areas of embedded software for which an AFLC has maintenance responsibility. Results to date indicate that no cost-collection mechanism is available for any of the functional areas and that no means to identify cost drivers for embedded software maintenance and operational support are available.

One local data base of reliable cost and schedule data has been established for the F-111 OFP maintenance facility. This data base provides budgetary estimates based on experience data. The instructions for each new release of the F-111 software total 400 and cost approximately $1 million. On the basis of the traditional cost-per-instruction-method for allocating budgeting funds, this staggering cost of instruction for maintenance of the F-111 OFP would not be believed. The data base is required for credibility.

28. D. A. Fisher, *Automatic Data Processing Costs in the Defense Department,* Institute for Defense Analyses, D/HC 73C0200, October 1974.

This report is very general and management-oriented.

29. T. Fleishman, *Current Results from the Analysis of Cost Data for Computer Programming,* System Development Corporation, TM-3026/000/01, July 1966.

Most of the specific data presented are now out of date.

30. E. K. Fong, *Unmanned Spacecraft Cost Model,* SAMSO TR-78-61, February 1978.

This cost model provides a parametric, cost-estimating relationship for unmanned spacecraft. Appendix C of this report, "Software Cost Estimating Relationships," presents an analytical methodology for estimating software costs, which requires only an estimate of the number of source instructions. The methodology used regression analysis of 25 software development efforts. Two relationships are presented for man-months (MM) per thousand instructions (I).

For real-time programs, MM = 28.899I. For support programs, MM = 62.557 + 2.170I.

31. J. E. Gaffney, Jr. and G. L. Heller, *Macro Variable Software Models for Application to Improved Software Development Management,* Workshop on Quantitative Software Models for Reliability, Complexity, and Cost, 9-11 October 1979.

This paper presents models of software productivity developed from empirical relationships between "macro variables" descriptive of the development process and productivity. It shows how these

models are used to improve costing of software developments. The paper also explains how these models can measure the impact of the uncertainty in estimation of program size on development schedule effort and cost. The models are based in part on earlier work by Norden and Putnam.

32. Clarence Giese, "Partitioning Considerations for Complex Computer Based Weapon Systems," *Journal of Systems and Software,* Volume I, 1979, 3-18.

The paper identifies criteria for partitioning a complex system into alternative data processing structures at the concept-formulation and contract-definition stage. This partitioning can minimize hardware and software life-cycle costs. The paper provides an analytical cost model for evaluation of alternative partitions.

33. C. A. Graver, E. E. Balkovich, W. M. Carriere, and R. Thibodeau, *Cost Reporting Elements and Activity Cost Tradeoffs for Defense System Software,* General Research Corporation, R. T. No. CR-1-721, March 1977.

This final report for the Air Force established certain variables (CERs) by using a large amount of data from a variety of postsoftware development projects. The report concluded that the CERs produced estimates that could be seriously in error depending on many factors beyond those identified in the CERs.

34. M. H. Halstead, *Elements of Software Science,* Elsevier Computer Library, 1977.

This classic book provides the basis of Halstead's considerable research into software science. Such attributes of a computer program as implementation efforts, clarity, structure, error rates, language levels, and size are derived from basic theoretical hypotheses. The results generally are more theoretical than most cost-estimating engineers will be comfortable with, but the results are important as the starting point of basic research in cost-estimating techniques for future developments in the state of the art.

35. M. H. Halstead, "Software Science--A Progress Report," *Second Software Life Cycle Management Workshop,* 21-22 August 1978.

This paper presents new results related to programming rates, error rates, and precision and accuracy of estimating relationships. It also gives equations for obtaining source lines of code per programmer man-month for any given job size.

36. M. Hamilton and S. Zeldin, "The Relationship Between Design and Verification," *Journal of Systems and Software,* Volume I, No. 1, 1979, 29-56.

This paper states that the design phase of the software life cycle must anticipate the requirements of verification and maintenance. All phases of the development process tend to be interrelated in that the output of one is the input of the next and may probably affect the resources required for its completion. This paper presents a checklist by which to analyze the technique or the methodology used by a particular project and relate resource allocations. Resource allocation is defined as the process that prepares one system (or phase) to communicate (interface) fundamentals affecting design, production, verification, and maintenance.

37. F. S. Ingragsia, "Combating the 90 Percent Complete Syndrome," Datamation, January 1978, 36-42.

This paper describes, from a management point of view, the problems in estimating progress in software development and in predicting future cost requirements.

38. Donald W. Kosy, *Air Force Command and Control Information Processing in the 1980s: Trends in Software Technology,* USAF Project Rand Report, R-1012-PR, June 1974.

This report forecasts software technology available in the 1980s for use in Air Force C^3 system applications. It also describes current state-of-the-art software technology and present and future costs.

39. V. LaBolle, "Cost Estimating for Computer Programming," *Proceedings of IEEE 1972 International Convention,* 20-23 March 1972, 28-30.

This paper contains 99 annotated references.

40. Ware Myers, "A Statistical Approach to Scheduling Software Development," *Computer,* December 1978, 23-35.

This paper explains a macro-estimating technique that generates an expected life-cycle curve of manpower against time. The technique starts by fixing the size, start date, and duration of each distinguishable activity. Adjustments are made for caliber of personnel, complexity, uncertainty, and other factors. More than 100 factors are identified. In the Doty study, 42 were significant. This paper gives an overview of several macro- and micro-estimation techniques.

41. P. V. Norden, "Project Life-Cycle Modeling," *Software Life Cycle Management Workshop,* 21-23 August 1977, 217-306.

The life-cycle model proposed is $y = k\,[1-\exp(-at^2)]$, where y is the cumulative man-months through month t, k is total manpower required, and a is a parameter to be determined. The life-cycle model is an econometric model describing the rate at which human effort is required to produce the software (more generally the rate at which complex problems are solved by humans). The model is used to forecast

man-months and time duration to produce the final software. Many tables and examples are given.

42. P. V. Norden, "Useful Tools for Project Management," *Management of Production,* Penguin Books, Inc., Baltimore, Maryland, 1980, 71-101.

Mr. Norden worked at the IBM Development Laboratory at Poughkeepsie, N.Y. He postulates that patterns of manpower buildup and phase-out rates in software development projects (and other complex production problems as well) are predictable. The Rayleigh curve is said to fit the observed rates closely. Equations for the curve and determining parameters are given in the paper. Putnam applied these results to software development to formulate the SLIM model.

43. "Panel on Software Estimating and Costing Approaches," *Proceedings, NSIA Software Conference,* Buena Park, California, 13-15 February 1979, 235-238.

44. R. R. Prudhomme, "New Software Methodologies and Quality Software," *1979 ASQC Technical Conference Transactions,* Houston, Texas, 357-371.

This report describes new software management, planning, and design methodologies for reducing life-cycle costs, errors, and schedule slippages. These methods manage software configuration as they would hardware, apply cost-effective quality techniques, and use top-down programming. The final goal is to reduce life-cycle costs by reducing testing, operation, and maintenance costs while delivering a higher-quality software product. It is stated that avionics software now costs $75 per instruction in development and $4,000 per additional instruction or change introduced in maintenance.

45. L. H. Putnam and A. Fitzsimmons, "Estimating Software Costs," *Datamation,* September 1979, 189-198.

This paper describes for managers how application software behaves, what factors can be controlled, and what factors are limited by the process itself. It presents results of five years of study of manpower vs. time for several hundred medium- to large-scale software development projects of different classes. These projects all exhibited the same life-cycle patterns of manpower requirements: a rise, a peaking, and a tailing off. One manpower curve and its describing equation are used to cost-estimate software development and maintenance effort.

46. L. H. Putnam, "The Influence of the Time-Difficulty Factor in Large Scale Software Development," *Software Life Cycle Management Workshop,* 21-23 August 1977, 307-312.

The paper develops quantitative measures to show why attempting to accelerate the performance of software development by adding more manpower is almost always unsuccessful. Large-scale software systems have been characterized by several cost overruns and time slippages from

schedule. This paper describes the relationship between productivity and manloading and gives equations that are useful in forecasting costs.

47. L. H. Putnam, "Progress in Modeling the Software Life Cycle," *Second Software Life Cycle Management Workshop*, 21-22 August 1978, 105-111.

An overview of a dynamic software life-cycle-cost model (SLIM) is presented. Productivity varies inversely as the square root of average applied manpower, so that efforts to speed up projects, increase productivity, and cut costs are nonproductive because these measures always are related functionally in a counterintuitive way. An example is given to show how to generate aswers to: How much will it cost? How long will it take? How many people? How much risk? What is the trade-off?

48. RCA/Government Systems Division, *Reference Manual,* PRICE Software Model, Version 3, PRICE Systems, Moorestown, New Jersey, 08057.

The PRICE Software (PRICE-S) Cost Model is RCA's empirical model for forecasting computer software costs and schedules. The model is designed to cover all types of software development, including business, communications, command and control, avionics, and space systems. Parametric techniques improve customary cost-estimating methods. In addition to cost and schedule estimation, PRICE-S provides automatic sensitivity and schedule effect analysis, together with monthly cost and progress summaries to support budgeting, risk analysis, and project tracking. The user must provide all necessary inputs.

49. *Rome Air Development Center Software Effort Data Base,* available from R. Nelson, RADC/ISIS, Griffiss Air Force Base, New York, N.Y. 13441.

50. V. B. Schneider, *A Unified Approach to Software Cost Prediction,* Report No. TOR-0078(3902)-3, The Aerospace Corporation, 24 May 1978.

The method presented is qualitative and general.

51. Jules Schwartz, "Resource Estimation," *Second Software Life Cycle Management Workshop,* 21-23 August 1978.

A statistical (regression) estimate for software resources required is RR = (S/A) C × P × P × L × I, where S is size in source lines of code, A is instructions per programmer day, C is a complexity factor, P is a personnel factor, L is a language factor, and I is related to debugging style. This highly subjective and empirical approach has a questionable basis in theory. There is no description of the measured error in using the model.

52. R. L. Smith, "Estimating Software Project Resource Requirements," *Structured Programming Series,* Volume XI, RADC-TR-74-300, January 1975.

The report presents a useful overview of available methods.

53. W. E. Stephenson, "Analysis of the SAFEGUARD Software Development Resources," *Proceedings of the Second International Conference on Software Engineering,* IEEE Catalog No. 78CH1317-7C, October 1976.

Data from several programs showing productivity versus size are given. Software is sized in thousands of instructions, and productivity is given in number of instructions produced per staff year and in hours per instruction. Three major complexity classes are identified. The paper concludes that for sufficiently large systems software can be sized in categories that permit productivity estimates as an explicit function of program size and complexity class. The complexity may be based on the design requirement as well as on the amount of code. The predictive equations are in the form, productivity = $A \times (\text{size})^B$ where the constants A and B depend on complexity and size.

54. H. Stone and A. Coleman, Life-Cycle Cost Analysis of Instruction-Set Architecture Standardization For Military Computer-Base Systems," *Second Software Life Cycle Management Workshop,* 21-22 August 1978, Atlanta, Georgia, 93-98.

This report describes a life-cycle-cost model for standardized-instruction-set-architectures using military computers. It shows that there is an inherent difference in value and utility of the existing support software bases of the several architectures. The commercially supported ones augment and maintain a substantial portion free of Government expenditure.

The report presents a method that identifies a course of action for computing the relative life-cycle costs for 78 Army/Navy hardware/software systems acquired over a 20-year period.

55. R. C. Tausworthe, "The Work Breakdown Structure in Software Project Management," *Second Software Life Cycle Management Workshop,* 21-22 August 1978, Atlanta, Georgia, 156-161.

The work breakdown structure (WBS) is a planning tool that links objectives with resources and activities in a logical framework. This paper summarizes the use of the WBS in software implementation projects and identifies problems in generating WBSs. Standard checklists of items to be included in the software WBS are given. The WBS provides the basis for estimating subtask durations, costs, risks, and constraints in software development.

56. R. Thibodeau and E. N. Dodson, "The Implications of Life Cycle Phase Interrelationships for Software Cost Estimating," *Second Software Life Cycle Management Workshop,* 21-22 August 1978, Atlanta, Georgia, 70-76.

This paper explores the hypothesis that software cost-estimating relationships must include the effects of resources allocated to all life-cycle phases and the impact of one phase upon the resources required in another. The objective is to obtain reliable estimates

of software life-cycle costs that are suitable for initial planning. This method requires empirical relationships between life-cycle cost and CERs. Project data were collected to substantiate the mathematical relationships. The CERs developed have relatively low precision and are applicable only to projects similar to those from which the data came. The major conclusion from this study was that the resources required for one life-cycle phase was critically dependent on what was used in the others.

57. S. L. Uhrig, *System Partitioning - Overview and Evaluation Model Validation,* BTL System Architecture Partitioning Study, Final Report to U.S. Army, BDMSCOM, Huntsville, Alabama, Defense Documentation Center, 1977.

Various analytic cost and productivity models are considered for each element of the software life cycle. Additional effort is required to refine these analytical models to develop techniques for estimating the coefficients that allow for all elements of the system. This study addresses maintenance and operation of the software in the system environment.

58. C. E. Walston and C. P. Felix, "A Method of Programming Measurement and Estimation," *IBM Systems Journal,* No. 1, 1977, 54-73.

A method of programming and estimation of project productivity is presented, together with preliminary results for measuring and estimating programming project duration, staff size, and computer costs. This method is directed toward measuring the rate of production of lines of code by project as influenced by a number of project conditions and requirements.

The paper describes the data reporting and analysis used in the software measurement programs of the IBM Federal Systems Division. The data base, the services available to users, and descriptive statistics from the data base are described. A programming productivity estimation technique is described in detail.

59. G. F. Weinweram, *Research Into the Management of Computer Programming: A Transitional Analysis of Cost Estimation Techniques,* System Development Corporation, TM-2712/000/00, Santa Monica, California, November 1965.

By relying on historical data from System Development Corporation on 74 software developments that use linear regression techniques to relate size and development time, the following equation is obtained $D = I(99.25 + 2.33\ I^{2/3})^{-1}$, where D is development time in months and I is number of instructions in 1,000s. The coefficient of determination (r^2) is seen to be 0.52. This relationship has several deficiencies if used as a forecaster of development time today. For example, the historical data are from the mid 1960s; any estimate of the number of instructions can be in error by 20 percent; and the source languages are now obsolete.

60. R. W. Woverton, "The Cost of Developing Large-Scale Software," *IEEE Transaction on Computers*, Volume 23, No. 6, 1974.

A cost-estimating algorithm is presented, and five traditional methods of software cost forecasting are described: top-down estimating, similarities and differences estimating, ratio estimating, standards estimating, and bottom-up estimating. This paper also identifies software information elements required for a valid cost data base. It describes two case studies that illustrate the software cost-forecasting methodology and historical results. Topics for further work and study are suggested. This paper is an excellent survey of general considerations and discussions relating to software development costs. It does not give specific techniques for estimating software life-cycle costs.

61. S. S. Yau, "Ripple Effect Analysis of Software Maintenance," *COMPSAC 78*, 13-16 November 1978, IEEE Catalog No. 78CH1338-3C, 60-65.

This paper analyzes the ripple effect that results from program modification. The analysis provides an important input for cost-estimating software maintenance requirements. A technique for analyzing ripple from both functional and performance perspectives is developed. A figure of merit is proposed for use in estimating the complexity of program modification; it can also be used to evaluate resources required. Rule-of-thumb estimates of software maintenance costs range from 40 percent to 70 percent of total life-cycle costs, a major consideration in any life-cycle-cost effort.

APPENDIX F

TI-59 SOFTWARE COST ESTIMATING MODEL

1.0 INTRODUCTION

The TI-59 program was developed as an implementing tool for ASD Cost Report Number 139, which describes a derivation approach to software costing based on RCA's PRICE-S (software) model. It is presented in two volumes. Volume I presents the core program and step-by-step directions for its use. Analysts should master the use of the core program described in Volume I before attempting the use of the more powerful and difficult options described in Volume II. Volume II presents several options that have been built into the program to enhance its usefulness and scope. Basically, the derivative approach uses PRICE-S to predict costs for varying numbers of instructions under a predefined development environment. These data are then used to fit representative regression equations. The most frequently used equations and work phase distributions can be entered as data for use by the TI-59 program. This allows a user unfamiliar with either PRICE-S or computer terminology to develop software cost estimates. Alternatively, any equation from Cost Research Report Number 139 or any other source, in like format, may be entered from the keyboard and used to estimate costs.

For the purpose of developing software CERs, Software Systems were modeled. Each Software System represents a complete software product, including tested programs and required documentation. Software product cost did not include the additional operational tests often required for weapon systems, general and administrative charges, profit, or integration costs. Cost for additional tests are not covered by this model. However, the latter three costs are accounted for by the TI-59 program. Software Systems are composed of subparts. A subpart is a general term for the next lower level of aggregation. A subpart could be a module, program, or sub-program. In addition, some of the data provided in ASD Cost Research Report 139 include an additional cost for integrating an assumed 10 subparts into the next higher aggregation level. Since the integration cost and 10 subpart assumptions may not always be appropriate, procedures are provided for eliminating the effects of these two assumptions.

Software System costs are assigned to three development phases—design, implementation, and test/evaluation. From a software product viewpoint, programming the actual coding or writing of the program only requires about 20 percent of the total software system effort. No adjustments have been made for the programming language (instruction set format) used other than to provide conversion factors between various languages and the machine language number of instructions required as input. In most cases at ASD, an assembly (type of language) level instruction is equivalent to a machine level instruction and occupies one word (a size unit) of memory in the computer. Thus, the terms instruction and word are often used interchangeably despite the technical distinction. A set of instructions (program) is designed to perform a given function. When this function changes, two areas may be affected: the program itself and the design work (documenta-

tion, etc.), both of which are necessary for a usable software product. Consequently, modifications of new design efforts are addressed by making adjustments in terms of the design/implementation and programming efforts which can be directly transferred from established software products. As more data become available, new software system equations will be developed.

2.0 DESCRIPTION OF MODEL

The model implemented by this TI-59 program was derived from RCA's PRICE-S software cost model. PRICE-S was used to predict the total direct and other labor man-months required to develop and deliver various numbers of executable machine level instructions in a given development environment. The man-month (MM) requirements under this scenario produce equations of the form $C(I)^e$ when regressed against the number of instructions (I). In this equation, C is a constant and e is an exponent. Errors (variations from actual PRICE-S outputs) are in the range of approximately plus or minus five percent. For reasonable ranges of I, the man-month value can be multiplied by a properly inflated cost per man-month figure to produce direct costs. An appropriate value for base year 1978 is approximately $60,000 per man-year. This value seems to be increasing slowly, perhaps 6 percent per year. To arrive at the total cost to the government, general and administrative changes (G&A) as well as profit are added. G&A values fall in a range from 5 percent to 15 percent of total labor costs, while profit averages 10 percent given the cost addressed in the $60,000/man-year input. The model also calculates the distribution of direct effort into three phases based on percentages representative of PRICE-S distributions for the scenarios modeled. The three phases are design, implementation, and test and evaluation. The model develops software system cost by calculating costs for individual software modules and the integration effort, based on user inputs. Software System modifications are taken into consideration by specifying the percent new design and percent new code.

3.0 PRICE-S CER VARIABLE OPTION DESCRIPTIONS

APPL—(Application) is a single parameter that summarizes mix of instruction types (real time, command and control, math, etc.) Typical values for airborne software fall in a range from 10.95 for all new design to 7.00 for major modifications.

RESO—(Resource) relates the scope of the work to the shop doing the work. Included in RESO are the effects of skill levels, experience, productivity, efficiency, and labor rates. The range on this value is from 2.0 to 6.0. A typical value for military projects is 3.5.

PLTFM—(Platform) relates the cost of the software development to the environment in which the software must operate. PLTFM is a measure of the reliability and testing requirements that must be satisfied. The range on this value is from 0.80 to 2.5. The typical value for mil-spec avionics is 1.8.

UTIL—(Utilization) is the larger percent of hardware speed or memory capacity used. Values of UTIL less than 0.5 denote no constraints on the system use and do not affect cost. Beyond this point, costs increase exponentially. The maximum input = 0.95. The typical value for mil-spec airborne avionics is 0.8.

CPLX—(Complexity) relates the difficulty of the task to the normal time required for this accomplishment. The standard input is 1. However, under unusual circumstances this value can be adjusted. For example, 1.2 would represent a hard application while 0.8 might represent an easy application.

INTEG—(Test and Integration) is the index of the level of test and integration necessary per software module to integrate it with a software system. The standard input is 0.5. Values range from 0 to 1.0.

INST—Using the simple approach, the number of instructions is the only necessary measureable input. This input should represent the number of deliverable executable machine-level instructions.

4.0 INPUT VALUES

CONSTANT—The equations used to predict man-months are of the form $C(I)^e$ where C is a constant, I is the number of instructions in thousands and e is an exponent. C and e values can be input by the user (see Vol. II) or selected from the matrix of built-in equations by specifying difficulty, a column identifier, and class a row identifier.

EXPONENT—(See CONSTANT).

INSTRUCTIONS—The value input is the number of deliverable executable machine-level instructions in thousands. For most airborne avionics the number of machine-level instructions and words are essentially the same.

G&A—General and Administrative charges are entered in decimal form. G&A normally ranges between 0.05 and 0.15.

PROFIT—This value is also entered in decimal form. The values average about 0.10.

PERCENT NEW CODE—This value is entered in decimal form. It represents the amount of code (program) which will have to be developed. The program uses this input to adjust costs to reflect modifications and developments which make direct use of previously developed code.

PERCENT NEW DESIGN—This value is entered in decimal form. It represents the amount of design effort that cannot be directly transferred from previous development efforts. It is used to reflect modifications and certain development efforts.

DOLLARs/MAN-MONTH—This value is multiplied times the number of man-months to produce dollar cost. Thus, whatever assumptions are built into this figure are reflected in the costs. For example, if the man-months cost represents an inflated value weighted by percent of work completed in a year, then the costs represent then-year costs.

DIFFICULTY—This value identifies a column in the program's equation matrix. In the core program examples, these columns (1, 2, 3) correspond to increasing levels of difficulty (easy, medium, hard).

CLASS—This value identifies a row in the program's equation matrix. In the core program examples, these row identifiers (0, 3, 6) correspond to different classes of software (Ground, Airborne, Space).

INTEGRATION—This is a value between one and ten. This value is translated to a percentage number which is added to the program cost calculations. An input value of five indicates typical or average amount of effort is required to integrate the module into the entire software system.

END

FILMED

1-83

DTIC